Concept and Design of the Hybrid Distributed Embedded Systems Testbed

Mesut Güneş   Bastian Blywis   Felix Juraschek
Computer Systems and Telematics
{guenes,blywis,jurasch}@inf.fu-berlin.de

Technical Report
TR-NO: TR-B-08-10

August, 2008

Institute of Computer Science, Freie Universität Berlin, Germany

Contents

List of Figures

List of Tables

List of Acronyms

AODV	Ad-hoc On-demand Distance Vector
API	Application Programming Interface
ARP	Address Resolution Protocol
BLOB	Binary Large Object
CST	Computer Systems and Telematics
DES	Distributed Embedded Systems
DSL	Domain Specific Language
DSR	Dynamic Source Routing
FPGA	Field Programmable Gate Array
ICMP	Internet Control Message Protocol
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
ISP	Internet Service Provider
LAN	Local Area Network
LUNAR	Lightweight Underlay Network Ad hoc Routing
MAN	Metropolitan Area Network
MANET	Mobile Ad-hoc Network
MIMO	Multiple-Input and Multiple-Output
NAS	Network Attached Storage
NFS	Network File System
NIC	Network Interface Card
OLPC	One Laptop Per Child
OLSR	Optimized Link State Routing

OS	Operating System
PoE	Power Over Ethernet
QoS	Quality of Service
RARP	Reverse Address Resolution Protocol
SSH	Secure SHell
TCP	Transmission Control Protocol
TETRA	Terrestrial Trunked Radio
TORA	Temporally-Ordered Routing Algorithm
UDP	User Datagram Protocol
USB	Universal Serial Bus
USRP	Universal Software Radio Peripheral
WARP	Wireless Open-Access Research Platform
WLAN	Wireless Local Area Network
WMN	Wireless Mesh Network
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network

Abstract

Wireless mesh networks are an emerging and versatile communication technology. The most common application of these networks is to provide access of any number of users to the world wide Internet. They can be set up by Internet service providers or even individuals joined in communities. Due to the wireless medium that is shared by all participants, effects like short-time fading, or the multi-hop property of the network topology many issues are still in the focus of research. Testbeds are a powerful tool to study wireless mesh networks as close as possible to real world application scenarios. In this technical report we describe the design, architecture, and implementation of our work-in-progress wireless testbed at *Freie Universität Berlin* consisting of 100 mesh routers that span multiple buildings. The testbed is hybrid as it combines wireless mesh network routers with a wireless sensor network.

CHAPTER 1

Introduction

1.1 Motivation

In the last two decades we witnessed the development of many different wireless networks, like *cellular networks*, *mobile ad-hoc networks*, *wireless sensor networks*, *wireless personal area networks*, and *wireless mesh networks*. In the first step, each of these kind of networks was proposed for a particular application scenario, for example cellular networks for voice communication and wireless sensor networks to gather and distribute data. Therefore, the research community studied each network on its own without regard to other ones. Now, the research community perceived that isolated networks are not as useful as interconnected networks.

The long-term goal of network research is to enable users to communicate from *any place* at *any time* with *anything*. This scenario is coined in the notion of *ubiquitous computing* or *pervasive computing* [1, 2]. However, there is no single communication technology nor communication metaphor which will realize this goal. Thus, we expect the integrated deployment of the aforementioned wireless communication networks. But right now, there is no full understanding of how to integrate and deploy these networks.

Although the architectures and intended applications of these wireless networks are different, they share some fundamental properties and thus possess similar problems that have to be addressed by the research community, for example *channel assignment*, *media access*, *mobility*, *routing*, *addressing* and *naming*, *locating the nodes*, etc. Of course, the same problems may show a differing weight in different wireless networks.

Nevertheless, we (can) consider them as one family of networks and study problems in their variance for different networks. The question here is how to study a problem appropriately. The research community follows different ways to study and develop network related issues, like analytical approaches, simulation, and experiments using testbeds. These studies and development environments aid the researcher in different ways.

As recent studies reveal [3, 4] simulation is used in most cases, since a simulation environment is cheap in terms of effort and complexity. As simulation studies do not provide a high degree of realism, due to simplifications in the models, the obtained results cannot be transferred into real world settings. In contrast, experiments using testbeds provide the researcher with a high degree of realism. But, particular studies of large-scale distributed networks are only feasible with simulations, since the costs and efforts to perform experiments in the real world are too high.

In this paper we discuss the hybrid wireless testbed at *Freie Universität Berlin* that is currently set up to accomplish holistic wireless network research. Our focus is on the testbed, because of our research geared towards real world applications and since many matured network simulation packages are available, for example ns-2 [5], OmNeT++ [6], Qualnet [7], and Opnet [8]. However, the design and management of a large wireless network testbed and the process of running experiments are not as well documented.

1.2 Goal of this paper

This paper has several goals which are given as answers to the following questions.

- I am a student and want to do my thesis in your work group. Do you have some introductory material or a tutorial?

  This technical report shall be a tutorial for students that want to do their thesis in our working group and thus need some introductory material. First of all, we introduce the main and most important concepts and notions of this research field.

  Please read Chapter 2, Chapter 3, and Chapter 4.

- I am a student and I have heard you are setting up a multi-transceiver mesh / sensor testbed. Why are you doing this and what are you trying to achieve?

  This paper describes the research context of the Distributed Embedded Systems (DES) testbed and the research fields we plan to contribute. We elaborate the reasons why we set up yet another mesh testbed and do not stick to the existing mesh technology in regards of protocols, etc.

  Please read Chapter 4.

- Which testbeds do already exist? In addition, what actual deployments of mesh and sensor networks exist and what do they provide or lack of?

  We give a short overview about existing testbeds and recent deployments. We present existing mesh and sensor networks, elaborate the intentions of the creators and maintainers. We explain for what kind of research or for what commercial purpose they are used for.

  Please consult Chapter 3.

- How can an open wireless testbed be set up?

  This report will provide a guide how to build a large heterogeneous wireless testbed. There are many pitfalls when a testbed is designed and realized. We discuss the essential components of a testbed and explain why they are required or useful. Components may be software as well as hardware.

  Please read Chapter 4 and our second technical report [9].

- How can I design and perform (long-term) experiments?

  We present a methodology to perform experiments on a testbed and discuss various aspects which have to be considered during experiment design and evaluation.

  Please read Chapter 5.

- I am a student preparing to do my thesis. What tools do you use for your experiments that are available on the mesh routers?

  We provide several standard tools in our customized Linux distribution. These include network management, traffic generation, and spoofing utilities. Additional software will be installed on demand.

  Please read Section 5.4.

1.3 Structure of the paper

The remainder of this paper is structured as follows. In Chapter 2 we introduce the basic concepts of wireless networks, clarify the terminology, and give a classification of networks which we consider throughout the paper. In Chapter 3 we present an overview of existing research and commercial wireless network testbeds. We also discuss their advantages and disadvantages and elaborate the differences to our testbed. In Chapter 4 we introduce the design and concepts of the DES testbed, discuss the architecture of the testbed, and describe the most important components. The accomplishment of a series of experiments in a testbed environment is labor-intensive and full of pitfalls. To minimize the labor and pitfalls we introduce a methodology in Chapter 5 to accomplish automated standard experiments. The paper closes with a conclusion in Chapter 6.

CHAPTER 2

Fundamentals

In this Chapter we introduce some terminology which is used throughout the technical report. Subsequently, we give a classification of wireless networks based on their most important properties and also discuss common application scenarios. Finally, we elaborate the properties of our wireless network testbed.

2.1 Terminology

A *host* denotes an end-device, for example a desktop computer, laptop, PDA, smartphone. Hosts are usually the *source* or *destination* of a connection. The terms *client* and *server* are synonyms for one kind of a host. A *mobile client* is a client which can change its location during operation. A *router* is a device which has route information and thus can forward data on the behalf of others, like hosts. A *mobile router* is a router which can change its location during operation. The term *mesh cloud* characterizes a group of mesh routers. These usually form a persistent backbone mesh network. A *gateway* is a device which connects two networks. For its operation the *gateway* has to be equipped with network interface cards (NIC) for both networks. It may happen that the gateway has to adapt the (control) data from the source network to the format of the destination network. A *node* refers to a device in the network, for example host, gateway, router. A *sensor node* refers to a tiny device with a processor, memory, and transceiver that has attached sensors. A *sink* is a sensor node which is always a destination and may provide gateway functionality to serve as interface to a different network type. Data are measured and sent to sinks by *source* sensor nodes. While the terms source and sink are often used in the context of wireless sensor networks (WSN) they can also be applied to other kind of networks.

2.2 Classification of Wireless Networks

The classification of wireless networks given in literature is ambiguous. Thus it is not obvious for which type of network a particular proposal is tailored. The specific network might not be the focus of classification but the application and task which has to be realized with it. In this section we elaborate the differences of familiar wireless networks based on some key characteristics.

2.2.1 Classification Keys

We use the following list of keys to classify wireless networks.

Number of nodes: This key characterizes the participating number of all devices in the network, like routers, gateways, or hosts. The larger the number of nodes in a network, the more difficult it is to manage the network.

Mobility: This key refers to mobile nodes in the network, for example mobile routers and mobile clients. A network with a higher degree of mobility usually exposes a higher dynamic topology. The degree of mobility will most likely affect other classification keys such as energy-awareness and data-rate, since mobile devices usually lack a fixed power supply.

Hop-Count: The number of hops between a source and destination. Networks with a higher hop-count usually do not rely on a stationary infrastructure and thus optimized routes. A high hop-count is likely to increase the latency of transmissions and decrease the throughput of a network.

Self-Organization: This key refers to the degree of human interaction required by a network, for instance for configuration and management. Thus a network with a higher degree of self-organization is a network which demands less human interaction.

Energy-Awareness: This key refers to the energy sensitivity of a network. A network has to be more energy-aware if the energy resource is finite.

Universality: Characterizes whether the network is tailored to a specific application. A network is more universal if it can be used for more applications. The opposite of universality is specialization.

Data rate: This key specifies the user-perceived throughput, for example the quality of a connection from a source to a destination. Usually, the higher the data rate, the better the connection throughput. However, this key has to be used carefully, since a wireless link may show low quality due to interference even with high data rates.

2.2.2 Wireless Personal Area Networks

A wireless personal area network (WPAN) provides wireless access in the ambience of a human user. It can be used to connect devices like smartphones with a headphone or to connect a PDA with a desktop. In the terms of our keys the number of nodes in a WPAN is small but may be moderate in some rare cases. Therefore, the hop-count will be low and one-hop communication dominates. A WPAN may move together with its user, while nodes themselves do not need to be mobile. Body-area or automotive networks are such examples. WPANs posses a moderate degree of self-organization, since the user will typically initiate connections to different devices. Thus, some configuration or authentication data has to be supplied by the user. A WPAN is also moderately energy-aware, because many devices will be in the vicinity of a device with unlimited energy resources, like a desktop PC or settop-box. A WPAN connection may be initiated for a particular application, but there is no limitation for a WPAN, thus it has a high degree of universality. The application scenarios may be limited by the available data rate which is also moderate in most cases. The transmission range is usually limited to several meters.

2.2.3 Wireless Sensor Networks

A WSN consists of a collection of sensor nodes. The network may comprise hundreds of thousands of devices, thus the number of nodes can be considered high. WSNs are intended for monitoring, data gathering, and data dissemination. Usually the sensor nodes are deployed once and then retain their position. There are also some WSN applications like animal monitoring in which the sensor nodes are mounted on the animals. In that case the WSN may show some mobility. As the main application is data measurement and gathering, the data has to be send to a sink node. Therefore, the hop-count may be moderate to high, depending on the network size and the overall topology. The amount of data sensed due to an event is usually small, thus they do not require a high data rate. As mentioned a WSN is very often deployed once and then left to itself. Thus, it has to deal with events like topology changes due to failing nodes and should have a high degree of self-organization. The lifetime of a WSN mainly depends on the efficient consumption of the limited battery power. Therefore, a WSN has to be highly energy-aware. Since WSNs will be deployed for a specific application they are not universal. They can be considered as specialized.

2.2.4 Wireless Mesh Networks

The notion of a wireless mesh network (WMN), while not uniformly defined, refers to a setup of so called mesh routers, each equipped with at least one NIC for radio communication. As an emerging technology aimed at multiple application scenarios no standardized definition has been formed yet. This fact partially originates from many different approaches of the research community, diverse application specific requirements, as well as the applied architectural design principles. A WMN usually comprises a high number of nodes. The clients of a WMN will be typically mobile, but the core or backbone, comprising the mesh routers, will not. Thus, we assume a moderate degree of mobility for a WMN. A communication partner of a mesh node may be inside the same WMN or in a different network, for example the Internet. Thus we expect a moderate degree in the hop-count. A WMN may comprise mesh routers from different operators. In the worst case each mesh router is operated by a different entity. Therefore, its degree of self-organization has to be moderate to high. Mobile clients also necessitate self-organization as they have to be handed over between mesh routers. A WMN is a universal network without application restrictions. The high data rate of a WMN supports all kinds of applications. A high degree of energy-awareness is not aspired.

2.2.5 Mobile Ad-hoc Networks

A mobile ad-hoc network (MANET) is usually defined as a temporal network formed by various entities in a particular situation. Thus, the number of nodes in MANETs may vary. As the term lets assume, the participating nodes will have a high degree of mobility and the topology changes frequently. Since the MANET is established for a particular purpose most of the connections will be inside the MANET. Connections to other networks rarely do exist. Thus a moderate hop-count can be assumed, especially considering the challenges of multi-hop routing in mobile networks. A MANET has to have a high degree of self-organization, since it is not tailored for a special application and participating nodes may fail or move out of range. Devices participate only a limited time in the network and may leave and join at will. The nodes in MANETs usually rely on a finite battery-based power supplies, but the temporal restriction of the network decreases the energy-sensitiveness. Thus, a MANET will show a moderate energy-awareness. A MANET is a universal network without application restrictions and provides moderate to high data rates.

2.2.6 Comparison

If we compare WMNs to WSNs, the nodes of the former ones are far more powerful. Since mesh devices usually rely on a fixed power supply, energy efficiency is not of prime importance to them. Sensor nodes in contrast must feature a high degree of energy-awareness because battery recharging or replacement may be impossible in certain locations. In many cases off-the-shelf hardware (CPU, RAM modules, etc) is used in WMN routers, in comparison to custom developed sensor nodes. Specialized components are rarely needed as the focal point is aimed at a general purpose network. In contrast a WSN is almost always tailored with a specific application in mind. Compared to WMNs, a WSN does not rely on high data rates since the traffic generated by routing sensor data from source to sink is very low.

The notions of WMNs and MANETs are closely connected. Depending on the point of view, a WMN can either be considered a special MANET or a MANET as WMN with mobile routers. The number of nodes participating in both networks can be considered of equal magnitude and is much higher than in traditional body-area networks, respectively WPAN. While nodes in MANETs are usually considered mobile, mesh routers forming the backbone of a WMN are stationary. Therefore, the degree of self-organization is much smaller in a WMN. Since mesh routers usually rely on a persistent power supply, energy-awareness is not an important factor. Thus, the available data rate is often higher than in MANETs.

Disregarding the various definitions some shared properties of WMNs and MANETs do exist.

- multiple nodes creating a wireless network, often called mesh cloud
- more meshed/higher node degree than in traditional networks
- usage of unreliable wireless links
- often concurrent usage of more than a single channel
- usually omnidirectional antennas used in a dense mesh cloud, directional ones seldom to connect far away nodes or clusters

Figure 2.1 depicts radar-graphs for the discussed wireless networks according to their evaluation of the introduced classification keys.

2.3 Application Scenarios of Wireless Networks

A multitude of applications for various scenarios using wireless networks does exist. In this section we discuss some of these typical scenarios and elaborate their specific requirements. For each application and scenario we consider the suitability of the wireless networks introduced in Section 2.2.

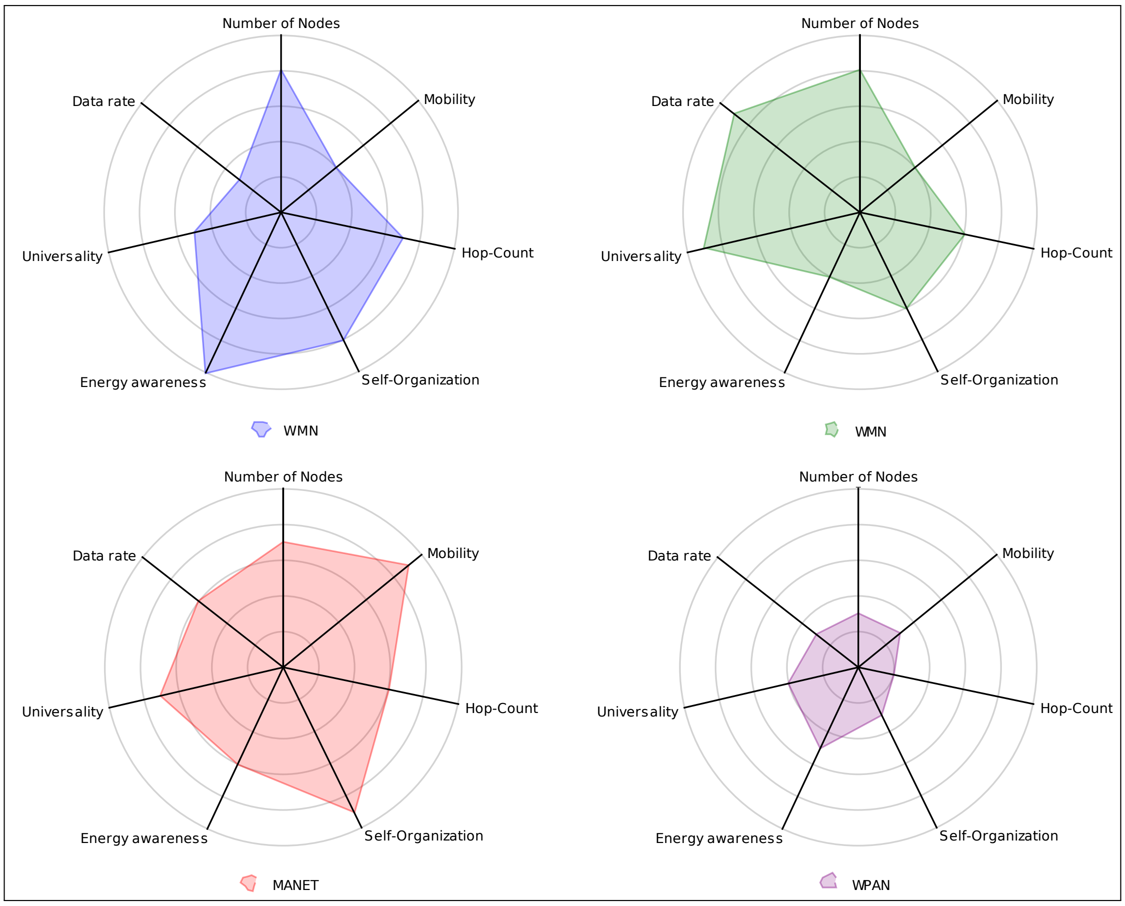


Figure 2.1: Comparison of different wireless networks. This figure depicts the evaluation of different wireless network types according to the introduced classification keys. The graphs show that the properties of the discussed networks according to the classification keys can differ by a significant amount. Therefore, advices which one to use for particular applications can be extracted.

2.3.1 Emergency

The emergency scenario is presumably the most cited one in literature. It describes the situation after a disaster of natural origin or caused by human beings when the normal communication infrastructure does not work - either partially or at all. Thus, a substitution has to be installed in the shortest time possible which should be adequate to support rescue operations. It is obvious that the installed network has to be simple to configure, easy to set up and maintain, and it has to adapt to a dynamic topology in order to support changes in numbers and density of participants. Voice and data communication are the most important applications for this scenario.

MANETs and WMNs can support this scenario. A MANET can be set up quickly but due to the dynamic topology, temporary isolated nodes without connectivity may be a result. Therefore, it is not as reliable as a WMN. In the case of a WMN the setup of the stationary router infrastructure is time-intensive and depends on a fixed power supply. As a benefit the installation of the backbone provides better reliability, throughput, and delay once functional. A WSN is not appropriate despite its high degree of self-organization and energy-awareness because the required data rates for voice and data transmission are out of range for sensor nodes. If the requirement of high data-rates is dropped, WSNs can be a viable option that has to be considered. This is especially true if text only messaging is necessary or other means of communication are still available, like Terrestrial Trunked Radio (TETRA).

2.3.2 Monitoring

The application field of monitoring comprises numerous variants, for example the monitoring of animals, habitats, elderly people, environments and surveillance. Depending on the *object of interest* which is to be monitored the requirements for the application can differ by a significant amount. In animal monitoring scenarios mobilility and energy-awareness are the most important features while temporary interruption of connectivity and high latencies are acceptable in many cases. For monitoring elderly people fast and reliable notifications are of primary concern in case of an emergency. High data rates and a reliable connections are especially required for surveillance scenarios with mounted video cameras that send video and audio data of adequate bitrate in real time.

With these different application fields and accompanying requirements outlined, the specific scenarios can be supported by a WSN, MANET, or WMN. Very often a combination of these networks will be most efficient. A WSN will be suitable for most animal monitoring scenarios, in which mobile sensor nodes mounted on animals may only be able to send gathered data once a day to a base station located inside the animal habitat. If data needs to be acquired more regularly, the deployment of more powerful nodes such as used in a MANET will be more efficient. Surveillance networks with fixed video cameras are best implemented with a WMN. The available high data rates and the reliability of the stationary mesh backbone meet the demands of such a scenario. In the case of monitoring elderly people it may be adequate to deploy a combination of a WPAN which monitors the patient at home and one of the other networks to transfer the data to a central station.

2.3.3 Community and Metropolitan Area Networks

The scenario is to provide network access, for example to the world wide Internet, for clients in a certain region. This area may cover a small rural village in developing countries or large facilities like company buildings or universities. The networks can be set up to connect workers or students and faculty members. In these days the most common usage can be found in big cities, in which commercial or alternative networks have been deployed for this purpose. The latter have been set up by groups which aim to provide network access independent of commercially operated networks. Advantages of such metropolitan area networks (MAN) rely in their independence of a wired infrastructure and in the ease for new members to join. Some regions also completely lack wired telecommunication facilities that otherwise would be too cost-expensive or even impossible to set up. In contrast to other network types, MANs often use off-the-shelf WLAN dongles or access points as the only necessary hardware required in order to access the network. Since the main goal of this scenario consists of ensuring Internet access, the network has to be very universal in order to support a variety of different applications. Real-time applications often require high data-rates and low latencies. In Section 3.2 we discuss some deployments of MANs.

A WMN is the most suitable alternative for community and metropolitan area networks. The stationary mesh backbone can provide reliable connectivity in the area spanned by the routers. It is also able to adapt to the network's growth with its increasing number of clients, thus supporting scalability in a high degree. Additional mesh routers can be deployed on demand in order to increase the available bandwidth in regions with high client density. The same applies to gateways whose number can be increased to reduce intra network traffic. Other network types, for example WSNs, are not suitable since the provided data-rates are not sufficient for this task. Further on, the hardware is often times limited regarding processing power and memory. The advantage of WMNs over MANETs lies in their reliability of the connectivity due to the stationary mesh routers.

CHAPTER 3

Related Work

In this chapter we review related work from the research communities as well as commercial ones.

3.1 Existing Wireless Research Networks

In this section we will discuss several existing wireless networks. Due to our approach to incorporate a wireless mesh and sensor network in a unified testbed we discuss a selection of well known setups of both kinds. In the following we differentiate between testbed installations that are usually indoor based and deployments for real world application scenarios. The latter ones may be indoor and outdoor based. Pure commercial deployments are excluded in this section but are discussed in Section 3.2.1. Testbeds are considered to be more universally utilizable than deployments as the latter are commonly installed for a particular cause. Research testbeds are usually set up as platforms for experimentation in a closed environment, shielded from hazards the devices are exposed to in deployments. The absence of these external, non-controllable factors does simplify and somehow partially abstract from real world scenarios. Nevertheless, testbeds allow to focus solely on the observation of network behavior. If needed, interfering factors may be generated artificially and observed network deviations might directly linked to these. We deem testbeds to be focused on research and development while deployments are centered on field testing and real world data acquisition. The latter operate as service provider in a production like environment.

3.1.1 IEEE 802.11 based Wireless Mesh Networks

Mesh networks based on the Institute of Electrical and Electronics Engineers (IEEE) 802.11 standard that are comparable to ours will be discussed as next. We focus on properties like the number of routers present in the network and the supported routing protocols. In addition, we are interested in the used operating systems and whether the WMNs support mobile routers. The last property we use to differentiate is whether testbeds do provide Internet access to arbitrary users.

Testbeds

The *Ad hoc Protocol Evaluation* (APE) Testbed is a project of the *Uppsala University* [10]. Their definition of "testbed" deviates from the general one introducted in this publication.

APE has to be understood as a software package distribution of an Operating System (OS) and various tools to evaluate ad-hoc protocols. This is a contrast to persistent testbed installations of routers that make up a mesh network. All nodes of APE are regarded as mobile without restrictions made to the movement. The area of movement may span several buildings. Due to the focus on the mobility and therefore MANETs, it is understandable that no infrastructure mesh network is present. The software distribution supplies a Linux kernel, two file systems (root and swap), and various tools including a MAC filtering program, an analysis frontend, as well as a graphical network viewer. In addition, a data gathering and logging facility is provided. At the time of the writing APE supports 6 routing protocols: Mad-hoc Ad-hoc On-demand Distance Vector (AODV), AODV-UU, Optimized Link State Routing (OLSR), Lightweight Underlay Network Ad hoc Routing (LUNAR), Temporally-Ordered Routing Algorithm (TORA), and Dynamic Source Routing (DSR). The vanilla APE distribution is currently limited to ORINOCO IEEE 802.11 WaveLAN cards. To attain reproducibility of experiments with mobile nodes so called scenario choreographies are introduced. Each of the laptop computers carried by a test person displays visual commands that instruct them what to do. Examples include the instruction to move to the end of a hall from the current position given a particular time or to remain in position and than continue walking. While the unbounded mobility of all routers differentiates APE from the other testbeds, experiments require large amounts of human resources (especially their time). Execution of a single time-consuming experiment spanning several hours or the repetition of shorter ones seems to be challenging.

The *Broadband and Wireless Network* (BWN) Lab at the *Georgia Institute of Technology* runs a testbed called BWN-Mesh [11]. 15 routers form the mesh network using IEEE 802.11b/g cards. Client devices are represented by several laptops and desktop PCs. Some of the routers act as gateways into the "next generation Internet testbed" located at the same faculty. The goal of BWN-Mesh is to study adaptive protocols for MAC, routing, and transport layers including their cross layer design. The testbed is integrated with the BWN sensor network testbed consisting of MICA motes running TinyOS.

Carleton University's wireless mesh network [12] uses routers based on Intel IXP425 series XScale processors that are equipped with two wireless interface cards. Of these Mini PCI NICs one is IEEE 802.11b and the other one is 802.11a/g compliant. The mesh routers are running μClinux - a Linux derivative that has been developed for MMU-less microcontrollers. No hard disk is present but 16MB of flash memory. A Quality of Service (QoS) enhanced version of OLSR is used as routing protocol for the network that is purely IPv6 based.

The *Experimental Computer Systems Lab* (ECSL) at the Computer Science Department of *Stony Brook University* did set up a testbed of 9 routers called *Hyacinth* [13]. These are small form-factor PCs (RouterBoard RB-230) running the Windows XP OS. Each of the wireless routers is equipped with three IEEE 802.11a network cards. A four node testbed using different hardware is also mentioned in [14]. We assume this installation to have evolved into *Hyacinth*. The focus of research has been on the topic of channel assignment and routing [13].

Purdue University's wireless mesh network testbed is called *Mesh@Purdue* (MAP) [15]. Small form-factor desktop computers (Pentium 4 based) running Mandrake Linux with Atheros chipset based IEEE 802.11a/b/g Mini PCI cards are used as well as Prism 2.5 Senao and Orinoco Wavelan PCMCIA cards. Some of the routers deployed outdoors are equipped with directional antennas. 5 laptops and 16 PDAs are used as mobile clients of the testbed. The four building spanning network of a total of 32 nodes is first and foremost used for Internet access. OLSR and an enhanced AODV version are available. Some of the published experimental results have been measured using MAP and in addition in cooperation with *Microsoft Research* on a second testbed.

Microsoft Research is working on a community mesh network [16] for the Windows OS. Their goal is to enable the setup of community mesh networks. Residents of a neighborhood shall be able to share existing Internet gateways. The *Mesh Connectivity Layer* (MCL) is a virtual network driver for Windows providing an interface to the mesh network. MCL is located between the MAC and network layer of the protocol stack. A modified version of DSR [17] called Link Quality Source Routing (LQSR) [18] is supplied as network protocol that uses MAC addresses for routing.

The *Miniaturized Network Testbed* (MiNT) at the *State University of New York* offers an approach for experiments involving mobility in close spaces [19]. It has been the core objective of design to keep every entity of the testbed in one room to reduce the effort of setup and management. For this, radio signal attenuators are installed between the NICs and the antennas. These limit the radio range by a constant non-alterable factor. The current version of MiNT consists of 12 mobile nodes [20] based on iRobot's Roomba vacuum cleaner and the RouterBOARD 230. Each router is equipped with 4 Mini PCI IEEE 802.11a/b/g NICs and antennas mounted on top of the enclosure. MiNT enables hybrid simulation by providing a way to conduct experiments in a realistic setting. To counter the lack of accurate physical layer simulation models the lower two layers of the simulation are replaced by the testbed hardware. Experiments for the ns-2 simulator can be carried out using real hardware by mapping a simulated node to each mobile router.

The *MIT Roofnet* [21] consists of around 50 routers deployed in apartments in near vicinity of the university's campus. The network is set up unplanned and is used for Internet access of students. At least three nodes use directional Yagi antennas while the rest is equipped with omni-directional ones. The routers are small form-factor PCs with hard disk, CD-ROM drive, and PCMCIA wireless local area network (WLAN) card in a PCI adapter. Participation and installation of routers in the project is done by volunteers. The network topology has to be considered as dynamic as they might join or leave at arbitrary times. Central to the project is the ease of setup so that even laymen can participate by extending the network and gaining access. Software updates are done over *Roofnet* itself without user interaction. The Click modular router [22] is used for route discovery and packet forwarding. *Roofnet* uses the SrcRR routing protocol that has been inspired by DSR to find high-throughput routes.

The *Berlin Roof Net* project [23] is inspired by the *MIT Roofnet*. Mesh routers are set up and run independently by students at *Humboldt University of Berlin*. The network is also used for Internet access. The routers run Click on the OpenWGT Linux distribution, which is a port of the well-known OpenWrt distribution to the utilized Netgear WGT634U access points.

The *Open Access Research Testbed for Next-Generation Wireless Networks* (ORBIT) is a two-tier laboratory emulator/field trial network testbed [24]. 64 routers are placed in a 8×8 grid with 1 m spacing in an indoor location. A planned extension to a 20×20 grid is also mentioned. This installation is used to create specific topologies for initial implementation evaluations. The field trial network spanning 10 km^2 that combines IEEE 802.11 and cellular (3G) radios shall enable real world like experiments. The routers are small form-factor PCs (Via C3 based) with two WLAN interfaces and local hard disk. A so called "integrated chassis manager" is used for remote monitoring, reset, and power on/off of the hardware. ORBIT offers an extensive software stack enabling experiment definition, execution, and visualization of measured results.

MeshNet is a testbed located at the *University of California, Santa Barbara*. A total of 25 routers are available whereas each of them consists of two Linksys WRT54G access points strapped together. The WRT54Gs are configured to different channels and are connected by Ethernet to each other. One of them participates in the mesh network as router while the other one is used as management interface that is accessible via an already present WLAN infrastructure. The OpenWRT Linux distribution for embedded devices is used as OS. A version of AODV with a reliability-based routing metric is available.

The *Ultra High Speed Mobile Information and Communication* (UMIC) testbed of the *RWTH Aachen University* pursues a hybrid approach that is different from the one mentioned above. 51 mesh routers based on ALIX.2C2/3C2 mainboards equipped with IEEE 802.11a/b/g NICs and several laptops are one part of the setup. A virtualization environment is also available for software development and functionality validation. Virtualization of routers is done by the Xen virtual machine monitor. The Generic Routing Encapsulation (GRE) tunneling protocol emulates a broadcast medium on top of Internet Protocol (IP) and connects all virtual hosts. DYMO and OLSR routing protocol implementations are available for both parts of the hybrid testbed.

The *Wireless Scalable and Efficient Mesh network* (WiSEMesh) testbed [25, 26] at the *Korea Advanced Institute of Science and Technology* (KAIST) consists of 56 nodes installed in 7 buildings. These provide Internet access to about 1000 users. Routers posses multiple NICs (at least two) that are connected via universal serial bus (USB). Two kinds of IEEE 802.11b/g cards are present - one Prism and the other one Ralink chipset based. The nodes are denoted as heterogeneous but are mostly small form-factor PCs (Intel Celeron with 1.5GHz, HDD) using Ubuntu Linux with a 2.6.x kernel version. OLSR has been chosen as routing algorithm. This testbed is also characterized as hybrid, but in a different context. The mesh network is mentioned to be used as backbone network for a WSN. No publication with further detailed information about this fact does exist.

Deployments

There are few non-commercial deployments of WMNs in this context that satisfy our definition as the WSNs in Section 3.1.2. The already discussed *MIT Roofnet* and *Berlin Roof Net* fall partially into this category as the projects set up mesh networks that are outdoor based and span more than just adjacent campus buildings. Their primary cause of installation in addition to research has been very application oriented as the network shall provide Internet access to various people. WiSEMesh can also be considered to be part of this group although it is restricted to campus buildings. The network is also set up by a particular single group. This is a difference to the above projects were arbitrary individuals can join and extend the mesh cloud.

The *Freifunk* (translation: free radio) community [27] is a non-commercial initiative based in Germany, Austria, and Switzerland. Their goal is to set up community area networks shared by all participants. Although not the primary cause of the individual projects, data about network usage, link stability, mobility, and various other properties are collected. Most noteworthy, the freimap tool [28] a visualization and analysis environment has been developed to generate statistics and display network topology maps.

Summary

We list the properties of the discussed projects that are shown in Table 3.1. This information has been compiled from the various publications of the WMN testbeds. Pure deployments had to be excluded due to their number. Thus, we focus on the following aspects of the discussed testbeds:

Project: The project name is either taken from the project description or the name of the university if the testbed is unnamed.

Nodes: The number of nodes in the wireless mesh network testbed. The values have to be interpreted with care as publications and websites often list deviating numbers.

Radio: The radio technology used for communication between routers. As we focused on IEEE 802.11 based mesh networks the technologies include IEEE 802.11a/b/g/n.

OS: The operating system used to run the wireless mesh network testbed.

Routing: We differentiate whether routing is done based on MAC or IP addresses inside the protocol stack. In addition we list which routing algorithms are available in the testbed and supported by the maintainers.

Mobility: If checked, all entities of the network can be mobile. No fixed infrastructure is present.

Internet: If checked, the network is used for Internet access of clients and therefore not a pure research testbed.

The number of routers participating in the largest mesh network is 64. As can be seen, the bigger testbeds have at least around 50 routers. With 100 routers targeted in the final setup phase of DES-Mesh we will have one of the largest testbeds.

The radio technology of the current testbeds is based on the IEEE 802.11a/b/g standards. There is only one case where only IEEE 802.11a is used as all others support b/g. While there is already IEEE 802.11n draft hardware on the marked, the newest radio technology based on Multiple-Input and Multiple-Output (MIMO) is not yet been used.

Most routers run a Linux distribution as OS- either for the common x86 architecture or MMU-less microcontrollers. The only exception are *Hyacinth* and the *Microsoft Research* testbeds. We suspect the availability of the full source code, absence of license fees, and therefore unproblematic code reuse and distribution as the main reasons. The decision for Windows XP in *Stony Brook University's Hyacinth* seems to be related to their research cooperation with Microsoft whose choice of OS is of course obvious.

Routing is mainly done based on IP addresses on the network layer. The prominent exception is the MCL by Microsoft. Not all projects name all of the available routing protocols. As we have learnt the *APE* testbed provides the most with 6 implementations while there are 2 AODV variants.

Project	Nodes	Radio	OS	Routing		Mobility	Internet
				Layer	Protocols		
APE	??	a/b/g	Linux	IP	AODV-UU, Mad-hoc AODV, OLSR, LUNAR, TORA, DSR	x	–
BWM-Mesh	15	b/g	??	??	??	–	–
Carleton Univ.	??	a/b/g	μCLinux	IP	QoS enhanced OLSR	–	–
DES-Mesh	100	b/g	Linux	IP	$\# \geq 10$	–	–
Hyacinth	9	a	Windows XP	??	??	–	–
MP	32	a/b/g	Linux	IP	AODV, OLSR	–	x
Microsoft Research	23	a/b/g	Windows CE	MAC	LQSR	–	–
MiNT	12	a/b/g	Linux	IP	??	x	–
MIT Roofnet	50	b/g	Linux	IP	Srcr	–	x
Berlin Roof Net	??	b/g	Linux	IP	??	–	x
ORBIT	64	a/b/g	Linux	IP	??	–	–
Meshnet	25	a/b/g	Linux	IP	AODV	–	–
UMIC-Mesh	51	a/b/g	Linux	IP	DYMO, OLSR	–	–
WiSEMesh	56	b/g	Linux	IP	OLSR	–	x

Table 3.1: Overview of wireless mesh network testbeds. We included DES-Mesh in its final stage of setup. Unknown values are depicted by two question marks.

3.1.2 Wireless Sensor Networks

Various wireless sensor networks do exist either as research testbeds or real world deployments. We are primarily interested in the number of devices participating in the networks and the maximum continuous lifetime. The lifetime is an important aspect, if not the most important, in the domain of sensor networks where devices have limited energy resources. Due to the sheer number of WSNs, we cannot claim completeness and therefore selected well-known testbeds as well as deployments.

Testbeds

In an installation of the *Colorado School of Mines* called *Casino Lab* [29] 52 Tmote Sky motes were mounted to the ceiling of a roughly 24.5 m × 12 m laboratory. 26 Tmote Connect Ethernet gateways connect the motes to the local area network and provide power supply via power over ethernet (PoE). Unfortunately, no information about experiments and their duration is available.

ETH Zürich's Deployment Support Network (DSN) [30] uses two parallel networks for research. A wireless backbone network is used to program, monitor, and manage the actual WSN. 60 nodes in total are present in the testbed setup. Half of them participate in the sensor network and the remaining in the DSN. A DSN-Server provides a client interface to communicate with the DSN. Each DSN-Node has exactly one wired connection to one sensor node of the attached WSN. A multitude of sensor node hardware is supported including BTnode, Tmote, TinyNode, and A80 target devices. Although TinyOS is supported, the BTnut OS is used with AODV as routing algorithm.

The largest known testbed is named ExScal [31] (*Extreme Scale Wireless Sensor Networking*), consisting of about 1000 eXtreme Scale Mote (XSM) sensor nodes and around 200 eXtreme Scale Stargate (XSS) backbone nodes. The investigation of the challenges of up to ten times as many devices has been the initial motivation of the DARPA funded Extreme Scaling project. Hundreds of thousand nodes are envisioned for future WSN applications. The backbone network uses IEEE 802.11b network cards and thus is also one of the largest ad-hoc mesh networks. The XSM nodes run TinyOS with Logical Grid Routing (LGR) while the XSS backbone routers run Linux. A custom beacon-free routing protocol called Learn on the Fly (LOF) is used for backbone traffic. Due to the sheer size, the testbed is located outdoors. We suspect the project to be discontinued or no active research being done, as no publications are available since the end of 2005.

Related to ExScal the *Kansei* [32] testbed uses 210 dual nodes, consisting of a XSM based sensor network and IEEE 802.11b XSS mesh nodes with a subset of 150 supporting IEEE 802.15.4. Five robotic mobile nodes are additionally available as well as a portable array of 50 Trio motes. *Kansei* has been initially conceived to test middleware services for *ExScal*. No information about conducted experiments and their duration seems to have been published yet. At the time of writing 106 nodes were listed as total number and only 61 in an up state according to the tier-2 health status website. This snapshot might not represent the actual state of the network but according to their web interface no experiments have been completed in the past weeks.

MoteLab [33] at *Harvard University* is a very popular testbed of 190 TMote Sky motes, scattered over 3 floors that are connected via Ethernet to the Local Area Network (LAN). The nodes run TinyOS. A quota based system enables the scheduling of experiments for registered users. To get the maximum duration of a single continuous run, extensive investigations would be needed. At the time of this writing, 93 nodes had been disabled either due to problems or manually. Of course this number depends on the current experiment and used firmware image.

The *Tutornet* [34] testbed at the *University of Southern California* is organized in 13 clusters consisting of 1 Stargate single-board computer mesh node (equipped with a IEEE 802.11b NIC) and several sensor nodes (91 tmoteSky and 13 MicaZ) that are attached via USB and run TinyOS. We found documented experiments of 3 to 12 hour duration.

Since 2005 the *TWIST* testbed at *Technical University Berlin* offers facilities for indoor deployment research. Its architecture uses the NSLU2 network attached storage (NAS) as super nodes for eyesIFX and Telos sensor nodes. About 200 of them are placed over 3 floors running TinyOS. We found data collected over 7 days in the corresponding technical report [35].

Deployments

In many cases WSNs are used to obtain field study information about animals and their environment. Unobtrusive devices spread over a large area are superior to traditional observation methods as some species are very sensitive to human disturbance.

One of the most famous and largest WSNs has been deployed as part of a study by the *University of California, Berkeley*. Initially in 2002, 32 Mica motes running TinyOS have been installed on Great Duck Island [36] for habitat and environmental monitoring. Until the year 2005 the network was extended to a total of 147 nodes [37]. A maximum continuous runtime is not mentioned.

In a similar joint project of *Microsoft Research*, *Freie Universität Berlin*, and *Oxford University* 10 MSB-430 modular sensor boards were deployed on Skomer Island [38] to observe the borrows of domestic birds. Due to the sufficient radio range the topology was a simple star making a routing protocol unnecessary. The test pilot network was deployed from March to June 2007. Data over a time frame of one month has been made available to the public. The current version of the network uses the MSB-H sensor nodes and the MicroMesh routing protocol on a customized ScatterWeb OS.

Princeton University's ZebraNet project [39, 40] equipped 7 zebras at Sweetwaters Game Preserve in Kenya with custom sensor node collars to collect fine-grained position data via GPS. The hardware is a custom product called ZebraNet node. One year of operation without human intervention has been set as an initial goal but to the best of our knowledge no long-term results have been published since 2004. It is one of the few WSN deployments with mobile nodes and therefore differs from the other ones mentioned in this section. Similar projects do exist or are proposed to monitor cattle[41], cats of prey[42], or various other animals.

The second large group of deployments uses scattered devices to monitor environmental changes. As mainly immobile phenomena have to be observed and monitored, sensor networks have the advantage of easier setup and data collection in remote locations versus traditional approaches.

The redwood monitoring *macroscope* [43] measured humidity, temperature, and photo synthetically active radiation data. Researchers from the *University of California, Berkeley* and *Intel Research* affixed about 27 sensor nodes to a 70 m tall tree. Mica2Dot, repackaged Mica2 motes produced by Crossbow, were employed running TinyOS and Tiny Application Sensor Kit (TASK) as software. MintRoute the default TinyOS routing protocol was used. Data was collected over a duration of 44 days.

Soil moisture and rain monitoring sensor nodes for remote Australian sites have been field tested [44] by the *University of Western Australia*. The hardware was based around Mica2 nodes running TinyOS. No routing protocol implementation was needed as the installation has been a single hop network. Depending on the used battery type, up to 16 respectively 30 days of lifetime have been observed at a 100% duty cycle. During rain periods more data per time unit are to be measured than in dry periods. The overall network lifetime is therefore decreased due to rainy periods. A later publication [45] showed actual data acquired over a 12 day time frame.

The *Colorado School of Mines* set up a network in the Edgar Mine [46] consisting of 10 nodes in a linear topology. At the time of this writing all sensors have been offline thus no data readings are available.

Volcano monitoring [47] field tests by *Harvard University* using infrasonic microphones have been performed with up to 16 sensor nodes and a 19 day span of time. The hardware was based on TMote Sky sensor nodes. A variant of MintRoute was run on TinyOS for status messages from the data sources while commands from the base station were flooded through the network.

In a roughly related application scenario the *GlacsWeb* [48] project of the *University of Southampton* used 8 custom designed sensor nodes called probes to measure a glacier's movement. But due to failing sensor nodes only long-time data of 3 probes could be collected. The maximum time frame has been 377 days in one case. The network's base station operated properly over about four months before it experienced power failure and needed maintenance. To our knowledge this seems to be one of the longest (continuous) published lifetimes of a deployment.

The PermaSensorGIS project [49] monitored various environmental parameters of permafrost in high-alpine regions in central Switzerland. A total of five sensors have been deployed which sent monitored data three times a day. The project's website [50] lists sensor node data for the time period of 27.11.2006 to 23.05.2007, i.e. roughly 6 months.

The *TU-Hamburg's* heathland experiment [51] used 24 Embedded Sensor Nodes (ESBs) for a field test with a two weeks scheduled runtime. The nodes ran a modified ScatterWeb firmware as OS.

Summary

We list the properties of the discussed projects that are shown in Table 3.2. This information has been compiled from the various publications and project websites. We focus on the following aspects of the discussed testbeds and deployments:

Project: The project name is either taken from the project description or the name of the university if the testbed or deployment is unnamed.

Nodes: The number of nodes in the wireless sensor network. The values have to be interpreted with care as publications and websites often list deviating numbers. This number includes all components like the sensor nodes as well as backbone routers.

Hardware: A list of all hardware platforms used and supported by the project.

OS: The operation system that runs on the sensor nodes. We do not list the OS of backbone components.

Routing: The routing algorithm(-s) mentioned in the project's publications. Many times no information is given but we assume MintRoute to be used in the cases where TinyOS is present.

Mobility: If checked, at least some entities of the network are mobile.

The number of nodes participating in the largest sensor network is 1000 with 200 backbone nodes in addition. As can be seen, most testbeds and deployments do not exceed the number of 150. Deployments are significantly smaller than testbeds with the Great Duck Island project being an exception. Although DES-WSN is smaller in size, 100 nodes are sufficient for sound research in our domain. Our research is not focused on scaling issues like in the *ExScal* project.

The installed hardware is very heterogeneous with platforms made by *moteiv/sentilla*, *ETH Zürich*, *Infineon*, *Freie Universität Berlin*, and *Berkeley*. The dominant OS is TinyOS by *U.C. Berkeley's* EECS Department. ScatterWeb, BTnut and Contiki are far behind. Our sensor network testbed is based on Contiki with a custom hardware abstraction and driver layer that partially originated from the ScatterWeb firmware.

Few routing protocols have been named in the publications. We suspect where no information is given and TinyOS is present on the nodes that MintRoute is used.

Mobility is a factor to be considered in only three of the projects. An evident disjunction between WSNs and MANETs seems to exist - at least according to our selection of projects. Mobility is therefore not a common property present in traditional WSNs.

Project	Nodes	Hardware	OS	Routing	Mobility
Casino Lab	52, 26	Tmote Sky, Tmote Connect	??	??	–
DES-WSN	2×100	MSB-A2	Contiki	??	–
DSN	2×30	BTnode, Tmote, TinyNode, A80, ...	BTnut	AODV	–
ExScal .	1000, 200	XSM, XSS	TinyOS	LGR	–
Kansei	2×210, 50, 5	XSM, XSS, Trio	??	??	x
MoteLab	190	Tmote Sky	TinyOS	??	–
Tutornet	104, 13	Tmote Sky, MicaZ, Stargate	TinyOS	??	–
TWIST	200	eyesIFX, Telos, NSLU2	TinyOS	??	–
Great Duck Island	147	Mica	TinyOS		–
Skomer Island	10	MSB-430, MSB-H	ScatterWeb	MicroMesh	–
ZebraNet	7	ZebraNet node	??	??	x
Redwood macroscope	27	Mica2Dot	TinyOS	MintRoute	–
Univ. of W. Australia	??	Mica2	TinyOS	??	x
Edgar Mine	10	??	??	??	–
Volcano	16	TMote Sky	TinyOS	MintRoute	–
GlacsWeb	8	unnamed custom product	??	single hop	–
PermaSensorGIS	??	??	??	??	–
heathland experiment	5	ESB	ScatterWeb	??	–

Table 3.2: Overview of wireless sensor network testbeds and deployments. The testbeds are listed in the upper part and deployments in the lower one. Unknown values are depicted by two question marks.

3.2 Commercial WMN Providers

In the last years hardware manufacturers started to develop devices especially tailored for wireless mesh networks. At the same time service providers evolved, which mainly focused on setting up municipal and community networks thus enabling network access for certain areas. Also some traditional ISPs showed efforts to create new possibilities for providing Internet access for their customers. The following sections tries to give an overview of the hardware used in commercial wireless mesh network set ups and of the current deployments.

3.2.1 Service Providers

Meraki

Meraki [52], an offspring of the *MIT Roofnet* project [53], offers home broadband Internet access via WLAN. The company sells custom hardware, in form of indoor and outdoor repeaters, which function as mesh routers. Backbone gateways with a wired DSL connection are installed on demand by Meraki in range of the mesh cloud. To join the network, the repeaters can be plugged into a standard notebook using a proprietary driver. The offered hardware and software, as well as the used routing protocol, are based on the technology developed in the *MIT Roofnet* project.

Meraki's hardware is available in three product lines, starting with the "Standard Edition", which is supported by advertising and thus subsidizes the hardware. Indoor repeaters of the "Standard Edition" are available for about $50. The "Pro Edition" ships with more features for network management and monitoring. Technical support is also provided. Finally, with the "Carrier Edition" the functionality to start a custom ISP service is provided. The edition includes a billing service and market analysis tools. No information could be obtained from online sources about the current number of subscribers or the number of deployed networks. Meraki launched the *Free the Net* project, which is located in San Francisco and aims at providing free Internet access for the whole city.

Mountain View - Google

Google teamed up with Tropos and Alvarion to deploy a mesh backbone providing free Internet access for the residents of Mountain View. In a first setup phase they installed about 350 Tropos mesh routers. 50 of those mesh backbone routers are connected to an Alvarion gateway. Internet access with a maximum data rate of 1 Mbps is free for any user with a valid Google account.

Other Service Providers

Recently, traditional Internet Service Providers (ISP) like US Internet and Earthlink cooperated with hardware vendors to set up mesh networks to enable wireless Internet access for customers. US Internet cooperates with Belair Networks for a project in Minneapolis, Minnesota where already 2000 mesh routers are deployed. Access fees for customers are comparable to wired connections. For more information the reader is also referred to Section 3.2.2 about Belair Networks.

EarthLink is another ISP from the USA involved in similar projects. By using Tropos mesh routers they deployed mesh backbones in several cities in the USA to enable client side Internet access via WLAN. EarthLink seems to have abandoned its mesh network in Philadelphia because only 6000 subscribers could be acquired for the service with a monthly fee of $20 and a bandwith of 1 Mbps.

3.2.2 Hardware Manufacturers

BelAir Networks

BelAir Networks [54] is located in Ontario, Canada and offers a variety of multi-transceiver hardware devices for a wireless mesh architecture. They mainly focus on municipal networks, but claim that their hardware supports a wider range of applications. It can be used to set up mesh networks for universities, armed forces, and governmental institutions.

BelAir Networks provides a variety of different mesh nodes each with up to six radios. Radio transceivers are available for 2.4 GHz WLAN, WiMAX, 4.4 GHz Military wireless networks, 4.9 GHz Public Safety (USA), and 5.9 GHz ITS support. In order to offer an efficient service each mesh node is equipped with at least one WLAN radio for client side communication and up to five additional radios to handle the backhaul traffic between the mesh routers. By using the IEEE 802.11b standard for client side communication, customers are promised a bandwidth of up to 6 Mbps which comes very close to the empirical maximum bandwidth achievable with this standard. They claim to provide this bandwidth by separating inner mesh network and client side traffic and avoiding possible interferences by using different radios and frequencies.

In recent projects BelAir Networks teamed up with local ISPs in order to create an infrastructure for city wide WLAN access. The biggest project takes place in Minneapolis, Minnesota where BelAir Network and the ISP US Internet are deploying a municipal network to provide Internet anywhere in the city. So far more than 2000 mesh nodes have been installed with the plan to attain a number of 3000 later this year. No information is given about how many of these mesh nodes are connected to the wired infrastructure. Therefore, the average Hop-Count and node degree is also unknown.

While BelAir Networks supplies the mesh routers, customers with WLAN enabled notebooks can subscribe at US Internet for a monthly fee of $19.99 for Internet access at data rates of 1 to 3 Mbps. Bandwidths of up to 6 Mbps are offered at $29 per month. These rates are on par to similar offers for wired DSL Internet access by other ISPs such as AT&T or EarthLink.

firetide

The company firetide [55] from California, USA offers a range of hardware devices for a three tier mesh network architecture. Their proposed mesh backbone network consists of HotPort 6000 Mesh Nodes, which can be equipped with up to 3 Ethernet ports and 2 radios. firetide claims to achieve a throughput of 70 Mbps for the mesh backbone. Their products support the frequency bands of 2.4 GHz WLAN, 4.9 GHz Public Safety (USA) and 5 GHz. The mesh protocol in use is a proprietary one called firetide AutoMesh. No further technical specification about this protocol is available to the public.

HotPort mesh nodes do route only inner mesh traffic. Client side traffic is handled by HotPoint 4000 Access Points, which can be mounted onto a HotPort mesh node via Ethernet. The third component, a HotClient 2000 Customer Premises Equipment, is used to integrate a far-away HotPoint with no direct connection to the mesh backbone. It does so by establishing a connection between the far-away HotPoint and one, which is mounted directly onto a HotPort via WLAN. Since the IEEE 802.11b standard is used, this limits the connection of the far-away HotPoints to a bandwidth of 5 Mbps.

firetide does also offer a management software to set up and maintain the network. Additionally, they provide the HotView Controller, a software component which manages the associations of WLAN clients to the HotPoints. This enables the network to handle fast roaming of clients from one HotPoint to another.

All devices are also available as consumer products usable for indoor and outdoor scenarios. The outdoor devices feature weatherized connectors and are therefore more expensive. Prices range from 300 to 500 € for HotClient CPE up to more than 2000 € for an outdoor HotPoint or HotPort equipped with 2 radios.

Recent projects show, that firetide's main field of application consists of outdoor video surveillance networks. Video cameras are mounted onto mesh nodes, which route the video data via the mesh backbone to a custom control station. During the Superbowl in 2008, a network of 42 cameras and 40 mesh nodes was deployed to monitor the crowd. Again, no information is given how many of the mesh nodes were connected to the wired infrastructure and to what degree a real multi-hop infrastructure was achieved.

Motorola MeshNetworks

Motorola MeshNetworks' [56] approach is very similar to the ones of the already discussed companies. It also relies on a separation of backbone mesh from client side traffic by using different radios or channels. Three product lines of mesh network devices are offered currently. For each line so called IAPs (Intelligent Access Point), wireless mesh routers, and custom PCMCIA modem cards for clients are available. They use the proprietary MeshConnex routing protocol that is described as proactive and reactive hybrid. No further detailed specification of MeshConnex has been published.

The MOTOMESH Solo line is equipped with one 2.4 GHz radio, which is shared for client access and node to node mesh links. The MOTOMESH Duo line mesh routers and access points can be equipped with up to two radios. In the single radio configuration, a 2.4 GHz radio is used both for client access and node to node mesh links. In the two radio configuration, a 5.8 or 5.4 GHz radio is dedicated for node to node mesh traffic, while the 2.4 GHz radio is used for client access. MOTOMESH Quattro access points and mesh routers can contain two IEEE 802.11 radios at 2.4 GHz and two custom Motorola mobile broadband radios, which can operate on 2.4 GHz or 4.9 GHz. One pair consisting of a WLAN and a broadband radio operates in the unlicensed 2.4 GHz band and in the licensed 4.9 GHz public safety band. Due to the redundancy a higher throughput is achieved but interference between the two pairs is very likely to occur.

Motorola's field of application is also specified as municipal networks and video monitoring. The most recent project took place in Los Angeles in 2007 and consisted of 10 video cameras, which were connected to mesh nodes. Motorala's devices and starter kits are also available on a consumer level. A MOTOMESH Duo starter kit consists of one access point and two mesh routers and is available for $7500.

Miscellaneous Hardware Manufacturers

Tropos [57] offers a range of hardware devices for a multi-transceiver mesh architecture. Their devices can be equipped with up to 2 radios to keep client and node to node mesh traffic separated. They make use of a proprietary routing protocol called Predictive Wireless Routing Protocol. Tropos is mentioned because they supplied the hardware of a recently installed mesh network that provides Internet access for residents in Oklahoma city covering 550 square miles. Unfortunately, no information is available regarding the network structure and if their topology is a multi-hop one. As mentioned above, they also supplied the mesh routers for Google's free Internet access project in Mountain View.

PacketHop [58] does also provide components for a mesh network architecture. Their implementation is meant to be standard-compliant to the IEEE 802.11s draft and thus already supports the Hybrid Wireless Mesh Protocol. Their mesh technology is available as a standalone embeddable closed source firmware module and as hardware devices like access points. So far there is no information about how these devices can be acquired, nor does the website list any information about ongoing projects.

3.3 Open Research Platforms

Next to the commercial development of mesh components, a couple of open research platforms evolved. These platforms were created either in an academic context or by the open source community and aim at aiding in research processes. In the following sections we introduce two different platforms and their suitability regarding the DES testbed.

3.3.1 WARP

The Wireless Open-Access Research Platform (WARP) of *Rice University* [59] aims at providing new research opportunities for wireless communication technologies by offering network devices which are programmable on the physical and network layers. The central unit of each WARP node is an Field Programmable Gate Array (FPGA) board which uses a Xilinx Virtex-II Pro and features four slots for optional daughtercards. So far three types of external daughtercards are available, the most important being the WARP Radio Board featuring a 2.4 and 5 GHz radio transceiver. Additionally, a WARP Analog Board with four analog inputs and 2 analog outputs is available next to a WARP Clock Board providing clock signals for WARP Radio Boards and for FPGA logic and digital converters.

The only drawback of the offered hardware lies in its price. All devices can be acquired in an commercial or slightly cheaper academic store. The price for a Starter Kit consisting of an FPGA board, two WARP Radio Boards and one WARP Clock Board ranges from $12,000 to $14,000. Thus, large testbeds with up to 100 nodes, like ours, are hardly affordable.

3.3.2 GNURadio

GNU Radio [60] is a free signal-processing software development toolkit under the GNU General Public License. The general idea is to enable research based on the electromagnetic spectrum without the need to acquire expensive radio devices. For this cause a software defined radio is offered. The time-critical signal-processing parts of GNU Radio are written in C++ because of performance reasons while most applications use the Python programming language.

GNU Radio is not meant to be just a simulation tool. Therefore, the Universal Software Radio Peripheral (USRP) open source FPGA board functions as an interface to the real world. It can be connected to a computer via USB 2.0 and extended by various external daughterboards to allow the USRP to be used on different radio frequency bands. Daughterboards are available for a couple of different frequencies, including 2.4 - 2.5 GHz used by WLAN. The USRP can be acquired for \$700. Optional transmitter and receiver daughterboards can be purchased for less than \$100 each while transceiver daughterboards can cost up to \$400. Although the devices are more affordable than the ones of the WARP platform, a testbed setup with up to 100 nodes can be created much cheaper with off-the-shelf components, which in contrast are not programmable of the lower levels.

CHAPTER 4

Concept and Design

There are several ways to study a wireless network related problem. Common environments are mathematical analysis, simulation, emulation, virtualization, and testbeds. There is no single environment to study a problem in all aspects. For instance, a testbed is suitable to study problems with respect to network performance but it is not suitable to study scalability aspects, since the number of available nodes may not be large enough. In that case it is better to use mathematical analysis or a simulative approach.

Since we are mostly interested in the practical and application specific aspects of wireless networks and a testbed provides the best environment for these kinds of studies we are building/constructing the DES testbed.

4.1 Motivation

The DES testbed pursues a hybrid approach. The nodes of our network are primarily wireless mesh routers using IEEE 802.11 network cards. Contained in the enclosure is a WSN node creating a second in parallel testbed. While the mesh part is called DES-Mesh the sensor network is named DES-WSN.

DES-Mesh is the core of the overall testbed providing a wireless infrastructure for sound research. As we discussed in Section 3.1.1 few of the existing testbeds offer an extensive hardware and software infrastructure. The number of routers is often below 50 and hinders the configuration of all kinds of topologies for experiments. Experiments using multiple different topologies are an important factor for research and to perform differentiated evaluations. From a software point of view we will provide the largest number of routing protocol implementations in a testbed. This includes the groups of proactive as well as reactive protocols. We will evaluate all of them with the help of DES-Mesh using various topologies. Standardized test cases will be defined for several application scenarios of WMNs. By using the Linux OS the outcome of our research will be available for others and the measurements revisable. Up-to-date implementations of protocols will be one of our contributions to the community. They are especially necessary as most of the discussed testbeds exist since several years. Thus, the software is often times outdated and does not compile against current kernel versions. As part of our research work we will review findings made with the help of simulation environments on a real world WMN testbed.

As mentioned, WSN are often intended to be applied in critical applications. These applications demand for *dependability*, which includes the concerns of reliability, availability, safety, confidentiality, integrity, and maintainability [61]. For typical industrial applications of wireless sensor networks these concerns have to be addressed at least partially, if not in total. To achieve this goal, different standards and techniques can be deployed, for example the IEC 61508 norm that is titled "Functional safety of electrical/-electronic/programmable electronic safety-related systems". As discussed in Section 3.1.2 there are only few deployments of wireless sensor networks for such critical applications. However, in future we expect to see them in increased numbers. Based on the publications none of the mentioned WSNs did run for a continuous time longer than a few months. It is left unclear whether these networks operated without any interruption or if partial or complete failures have been experienced, so that one has to suspect a much lower runtime. The published times therefore have to be skeptically scrutinized. We especially miss comparable information about failures and their effects on the whole network, respectively the overall application. Furthermore, from a differentiated point of view, the long running deployments have been very small as the average number of participating nodes has been way below 50. Their topologies thus were simplistic with a few exceptions. These networks differ considerably from the envisioned vast multi-hop wireless sensor networks used for monitoring of a multitude of phenomena over an extended time frame.

As conclusion we aspire the hybrid testbed setup called DES testbed. Roughly 100 nodes are anticipated in total to be able to create non-simplistic topologies. The continuous runtime of single experiments is targeted up to several months for the WSN part while DES-Mesh experiments will be of shorter time span. In the remainder of this chapter we will describe the hardware in detail, explain our overall architecture, and state the reasons to choose the particular components in our evaluation process.

4.2 Research Objectives

The DES testbed at the *Freie Universität Berlin* is a non-commercial research testbed with its focus on WMNs and WSNs. Figure 4.1 depicts the weight of the different classification keys as elaborated in Section 2.2.1 for these network types.

Many approaches to study WSNs and WMNs exist. Primarily, our research focus lies in experimental comparisons of simulation and real world test runs. Therefore, we decided to set up a testbed which allows experimentation in an environment close to the real world. As many publications have shown [3] [62], there is still a severe difference in the results obtained in simulation environments. In most cases deviant assumptions of the real world application scenarios, insufficient models, or specialized approaches focused on just a few details are to blame. Usually software solutions for many problems evaluated by simulation runs can not be transferred onto real hardware because of software technical reasons. Many simulators do not only abstract the wireless medium but also do not consider an OS used on real hardware. The properties of the OS can influence the experimental results because of the impact of critical parts, like scheduling strategies and multithreading. Another issue hindering a port of software components implemented in a simulator to the real world is the architecture of the protocol stacks. The developer is not bound to any restrictions implementing in a simulation environment while he has to use the provided Application Programming Interface (API) and features of a kernel that may have been developed over a extensive time span and more or less evolved than being planned from the beginning. We discuss these OS related issues in detail in [9].

An important aspect of future networks will be the integration of devices using several different radio techniques into one all-embracing network, such as the convergence

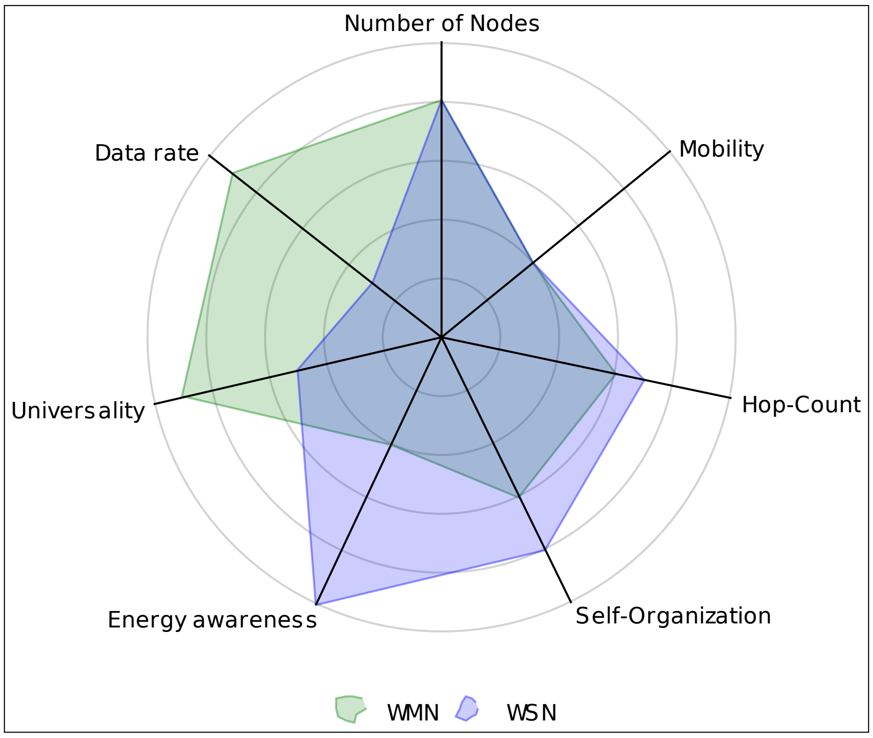


Figure 4.1: Comparison of WMN to WSN. The DES testbed will consist of two overlaying networks, a wireless mesh network and a sensor network. The figure compares the classification keys as introduced in Section 2.2.1. Both networks have similar properties in regard of the number of nodes and mobility. Sensor nodes have a higher degree of energy-awareness due being powered by batteries, while mesh routers usually rely on a fixed power supply. This has also an effect on the used radio devices and their transmitting power. Thus, a WMN can provide much higher data rates and is more universal in regard of supported applications.

of IEEE 802.11 with other technologies, like UMTS, Bluetooth (IEEE 802.15), WiMAX (IEEE 802.16), ZigBee, or Wibree. New radio technologies, media access protocols, applications, services, and topologies require novel approaches for routing protocols to facilitate an ubiquitous network.

Next to these research opportunities the wired connections between sensor nodes and mesh routers yield further novel research activities. The wired connection between the sensor node and the mesh node allows to conveniently manage the sensor nodes, while the WMN can make use of the sensor node as equal device providing additional channels next to the used IEEE 802.11 b/g devices. The approach will be described in more detail in [63].

Finally, the DES testbed will also be used in daily teaching giving students the opportunity to apply their knowledge attained in classes as well as working on assignments which can evolve to a bachelor or master thesis.

4.3 Testbed Architecture

Mesh routers and sensor nodes are deployed on our campus in a planned but not uniform manner. At the time of the writing we are setting up a pilot network in our institute building, spanning over 3 floors. In future extensions adjacent buildings will be covered.

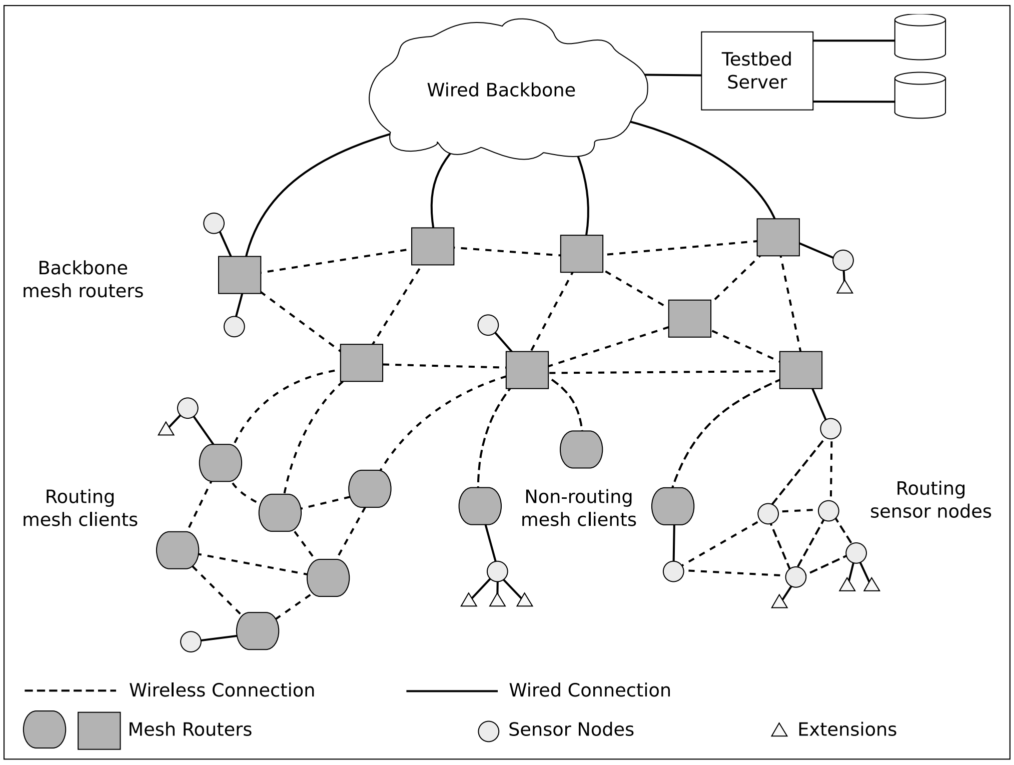


Figure 4.2: Architecture of the hybrid testbed consisting of mesh routers, mesh clients, sensor nodes, and management component. The sensor nodes may optionally be connected with wired extensions.

We discuss the placement in detail in Section 4.5. As depicted in Figure 4.2, our general system setup is a three tier based architecture. The stationary mesh routers build the core backbone network and represent tier 1.

Tier 2 comprises the mesh clients connected to the mesh backbone. The common mesh client is regarded as mobile and may offer a routing service depending on the configuration of the current experiment.

Tier 3 encompasses the sensor nodes which are connected by wire to a mesh router or client. A subset of sensor nodes may be operated independently of the mesh network infrastructure with alternative power source, creating a mobile and dynamic topology.

As we discussed in [63] the components of a hybrid tested can benefit from each other. WSN and WMN testbed shall not only coexist next to each other but exploit the features provided by the dual approach. We will integrate the MSB-A2 [64] WSN node because of its radio transceiver into the Linux kernel as a driver module. The sensor node shall provide an additional network interface. In the context of the ISO/OSI reference model the sensor node represents the lower two layers (physical and data link). This feature will offer us additional channels in the ISM frequency band that are orthogonal to IEEE 802.11. Recent reports [65] and our own measurements [9] have identified this aspect as one of major concerns for multi-transceiver based research. Interference does exist even between orthogonal channels.

The testbed infrastructure can also be fully exploited to establish virtual connections between far away sensor nodes using the Ethernet backbone. Whether WSN nodes function as routers is left to the application. The wired connection provides us with an additional reliable packet loss detection mechanism if packets are sent duplicated to the destination using the backbone. We deem this approach better (and cheaper) than a second in parallel wireless sensor network as the one discussed in Section 3.1.2. Unlimited power resources and data storage are of course properties being absent in real world settings of WSNs. To relax the first simplification, we intend the power consumption to be measured and nodes "shutoff" if a set threshold is reached. Like our previous hardware generation the WSN nodes can be equipped with a multitude of extensions. These sensors or actors may either be directly connected to the sensor nodes or to the mesh routers. In the latter case the sensor node uses the mesh router as relay. Our current research projects have taught us, convergence of WSNs and other technologies is important for industrial adoption and to solve real world problems. Therefore, we aspire full integration of WPANs, TETRA, and cellular networks like GSM or UMTS into the hybrid testbed.

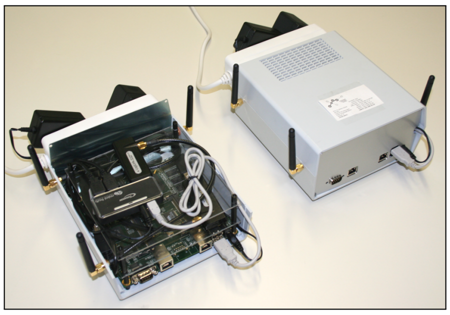
4.3.1 Backbone Mesh Components

The DES-Mesh routers are based on the PC Engines Alix2c2 embedded PC board [66] with a size of 152.4 × 152.4 mm as central component. The Alix family is the official Wireless Router Application Platform (WRAP) successor as the previously used AMD SC1100 CPU is no longer in production. CPU and memory are sufficient for our goals, removing the mesh router hardware from the list of possible bottlenecks in experiments.

Each mesh router is equipped with three or more IEEE 802.11b/g NICs connected to a small powered USB hub. We use dongles based on the RT2501U [67] architecture with a RT2571W BB/MAC IC and RT2528 RF IC. LogiLink WL0025 cards have been chosen because of their on-board R-SMA connector and included 4 dBi Hi-Gain antenna. The antennas are mounted at the side panels of the router using extension cables. Mini PCI cards have been disregarded as an alternative as we need at least three interfaces per mesh router and products with multiple transceivers are of limited availability. We discuss our choice of wireless NICs in Section 4.6 in detail. Future upgrades, for example once IEEE 802.11n has passed standardization, are easily accomplishable because of the usage of USB based devices. The second USB port connects the ScatterWeb MSB-A2 sensor nodes that will be discussed in Section 4.3.2.

The primary Ethernet port is used to boot the operating system over network and to mount the root file system while the second one remains unused at this time. Mesh routers have no need for local persistent memory and store everything on a central server. The enclosure is a customized TEKO AUS23 (198 × 178 × 90 mm) as the original Alix enclosure provides enough space only for the mainboard. Keeping everything in a single, small form factor, and easily handable package has been a core objective and excluded alternative solutions, like the NSLU2 NAS. We need at least three USB WLAN sticks, one sensor node, up to two USB hubs, and various cables to fit inside. Two intermediate levels keep these components separated from each other.

We have two options to supply the mesh routers with power. As for now, all mesh routers are powered by connecting a multi-plug AC adapter with the power supply system of the university buildings. Due to space and heat constraints the power supply is mounted to the back of the router. But this approach is not feasible for every placement since some locations lack access to the power supply system. Thus, we will make use of PoE adapters. The network infrastructure in our university building provides PoE-capable switches, which can be used for this matter. Using PoE adapters offers us several advantages. Rebooting unresponsive nodes can be achieved remotely by deactivating and reactivating the specific Ethernet port of the switch. This will be useful if a node can-


(a) Hybrid DES testbed node

CPU	500 MHz AMD Geode LX800
DRAM	256 MB DDR DRAM
Ethernet	2 Ports (Via VT6105M)
Expansion	2 Mini PCI slots dual USB 2.0 port
Storage	CompactFlash socket
Enclosure	customized

(b) Alix2c2 Specification

Figure 4.3: DES-Mesh Hardware Specification

not be accessed via Secure SHell (SSH) anymore and manual on-site handling becomes necessary. Therefore, depending on the experience gained once a larger number of experiments has been performed and a network size of 100 nodes is reached, equipping all mesh routers with a PoE power supply will be a subject of consideration. The whole package and Alix2c2 specification is shown in Figure 4.3.

By using multiple network interfaces our mesh routers are so called multihomed hosts. Multihomed hosts use more than one network interface. The term multihomed also refers to hosts having multiple layer 3 addresses configured to a single NIC. This is often done when using virtualization environments and the virtualized OS needs network access with a separate IP address. As an disadvantage the network bandwidth has to be shared. The different existing multihoming variants are as follows:

- Multiple NICs, each with a single IP address
- Single NIC, multiple IP addresses
- Multiple NICs, each with multiple IP addresses
- Multiple NICs, all share a single IP address

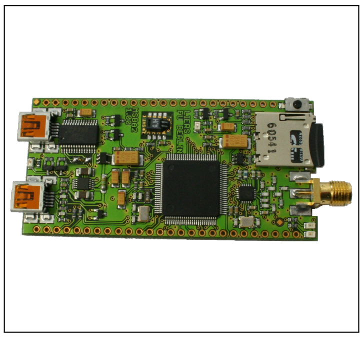
Any of these configurations can be used in experiments executed on our testbed on demand. As discussed in Section 3.1.1 only few testbeds provide such a feature.

A testbed installation requires a customizable and open OS. Linux fits our needs as it is already used in our day-to-day research. In contrast to other testbeds we will make use of the most recent kernel and driver versions and will keep up with the developments of the community. This is an important fact for real world application focused research. We noticed that several publications presenting results of WMN experiments did not use up-to-date kernel trees for their implementations or used obsolete features and interfaces[1]. We discuss our choice of hardware with regard to the OS in Section 4.6.

4.3.2 Sensor Network Components

ScatterWeb [68] nodes are connected to each mesh node via the USB port that provides unlimited power supply and is used as flash and serial terminal interface. Having the feature to write firmware images to the flash memory over USB has been one important

[1]This fact refers to the newest kernel version at the time of publication.


(a) Modular sensor board MSB-A2

Microcontroller	NXP Semiconductors LPC2387
CPU Frequency	up to 72 MHz
RAM	98 KiB
Flash	512 KiB
Transceiver	Chipcon CC1100
Expansion	GPIO pins mini USB 2.0 port
Storage	microSD-card socket

(b) MSB-A2 specification

Figure 4.4: DES-WSN Hardware Specification

issue, as it prevents unresponsive nodes that need manual on-site handling if an over-the-air flash process fails. The ScatterWeb nodes are called MSB-A2 [64] because their CPU is in contrast to our previous generations no MSP430 made by Texas Instruments but an ARM7 and the board is our second revision. We use an LPC2387 made by NXP Semiconductors. The 32-bit ARM7 TDMI-S core based microcontroller can be dynamically configured at runtime up to 72 MHz depending on the sensor network application and energy requirements. The Chipcon CC1100 transceiver in our configuration uses the ISM band at 863 to 870 MHz with a maximum data rate of 500 kbit/s. The MSB-A2 therefore sets up a separate in parallel testbed that is fully orthogonal to the frequency band used by WLAN.

The sensor node is equipped with a Sensirion SHT-11 temperature and humidity sensor. By using either simple general purpose input/output pins (GPIO) or an on-board mini USB port arbitrary extensions can be connected to the MSB-A2. In addition to the internal 512 KiB flash memory microSD-cards can be used. Virtually unlimited data storage may be accessed via the Ethernet backbone if necessary. The sensor network hardware specification is shown in Figure 4.4.

4.3.3 Management components

Management of the DES testbed is not an easy task due to the large amount of hybrid nodes. We introduce the *testbed server*, a central entity in the DES testbed, which provides various services for the hybrid nodes. The *testbed server* hosts the file systems via network file system (NFS) for the mesh routers over the Ethernet connection. Additionally, it hosts a PostgreSQL database, which is used to store logging data and measurement results obtained from experiments. Next to these infrastructural services, we require management components with which we can conveniently handle the DES testbed and simplify the execution of scheduled experiments. The main tasks of these management components are as follows:

- **Monitoring**: The state of the network needs to be monitored at all times, so that problems, like unresponsive nodes, can be discovered on time. If exceptions are encountered during an experiment, the test run might have to be aborted. Logging data has to be made accessible to the maintainers and users of the testbed in order to detect such erroneous behavior.

- **Configuration**: Since we do want to study a variety of applications using the testbed, different experiments require specific settings for each of the participating nodes. Therefore, the management component has to be able to configure groups of nodes prior and during an experiment.
- **Scheduling and execution of experiments**: Experiments have to be scheduled so that two separate experiments do not overlap. Additionally, the experiments have to be executed, which comprises many different tasks including preparation of the testbed for an particular experiment and making results and logged data accessible to the testbed user for further evaluation. Our approach to the experiment methodology is discussed in detail in Chapter 5.

The *testbed server* of our DES testbed represents the central instance of our management components and takes care of the listed tasks. The core of the testbed server consists of a web based application implemented in Java. It comprises three integrated software components:

DES-EXP: This software provides an experiment manager which is responsible for the scheduling and execution of experiments. Experiments according to our experiment definition as explicated in Chapter 5 can be created and scheduled to be executed. DES-EXP features a sophisticated user management module to ensure simultaneous multi-user access to the testbed, which will be convenient once the testbed will be used for lab exercises.

DES-CONF: Our network configuration tool is based on SNMP. It provides features to retrieve the state of the network and allows the users to alter the configuration of the testbed nodes.

DES-VIS: This component is a visualization software based on the JavaView framework. DES-VIS displays gathered data obtained from experiments. It features the ability to replay whole experiments or parts of them while showing additional information, such as the current links between nodes.

These three components are integrated into one web application hosted on a Tomcat server on the *testbed server*. It is accessible from within our university network, so students and research staff can conveniently use the provided interface to the testbed. In Chapter 5 we explicate how experiments can be performed using the discussed management components. Furthermore, we list standard tools that are provided in our customized Linux distribution and can be used in any experiment.

4.4 Network Management

Mesh networks can be classified based on the type of management, It is common to differentiate between fully and unmanaged networks, with semi-managed networks in between [69].

In fully managed mesh networks the routers are under control of a central entity. Clients form a separated group of non-routing nodes that connect to routers only. The clients do not need any special configuration as they do not participate in the core mesh network and require a gateway only. Topology-wise they are “leafs of the mesh cloud” and access to foreign hosts through the WMN is fully transparent. This is comparable to common IEEE 802.11 infrastructure networks with access points. In a typical real world installation routers would be setup by an ISP offering customers access over a wireless connection.

In contrast, an unmanaged mesh network is very similar to a MANET. Infrastructure can be missing in total and nodes, each managed by an individual, participate for an undetermined time in the network. As they can provide routing services and are usually mobile, links and routes are typically unreliable. The One Laptop Per Child (OLPC) project[70] for example enables children in developing countries to setup such networks for education purposes.

Semi-managed mesh networks fall in between these two groups. Clients are an integral part and can perform routing processes in cooperation with infrastructure routers that are central managed. Due to the access to the mesh network offered by additional nodes the covered region can be extended. A denser node population often leads to a higher node degree and eventually to more reliability because of redundancy. The topology changes more or less frequently depending on the mobility of client.

DES-Mesh is made up of the stationary core network routers that will always be present. We plan to incorporate mobile mesh components that will either be used as roaming clients or as mobile routers. Due to the properties of the hybrid testbed DES-WSN will also have a persistent stationary infrastructure. The 100 sensor network nodes inside the mesh routers will be supplemented by mobile and battery powered devices. All entities will be managed and administered centrally. Considering the above definitions the DES testbed is a fully managed network that deviates from the traditional notion because of the managed mobile entities.

The testbed server introduced in Section 4.3.3 provides the interface to the WMN as well as the WSN. We discuss in Chapter 5 how experiments are to be defined, managed, and evaluated using this central facility.

4.5 Setting and Node Placement

The DES testbed will span several buildings located on the campus of the *Freie Universität Berlin*. As shown in Figure 4.5 we plan to cover up to 6 buildings in its final setup phase. In the initial stage the computer science building will be equipped with a pilot network of about 35 mesh routers located on all three floors. The adjacent buildings are of similar height with at least one floor underground and two or three above. The total number of routers needed to span all buildings depends on many factors like:

- Desired node degree
- Attenuation factor of the buildings' infrastructure
- Interference due to external non-controllable sources
- Moving objects like people leading to undesirable short-term fading effects

The routers will be deployed in a non-uniform manner but to meet our requirement to create a topology that incorporates as many facets of real world installations. The following list does include some of the most interesting properties, but cannot capture all aspects:

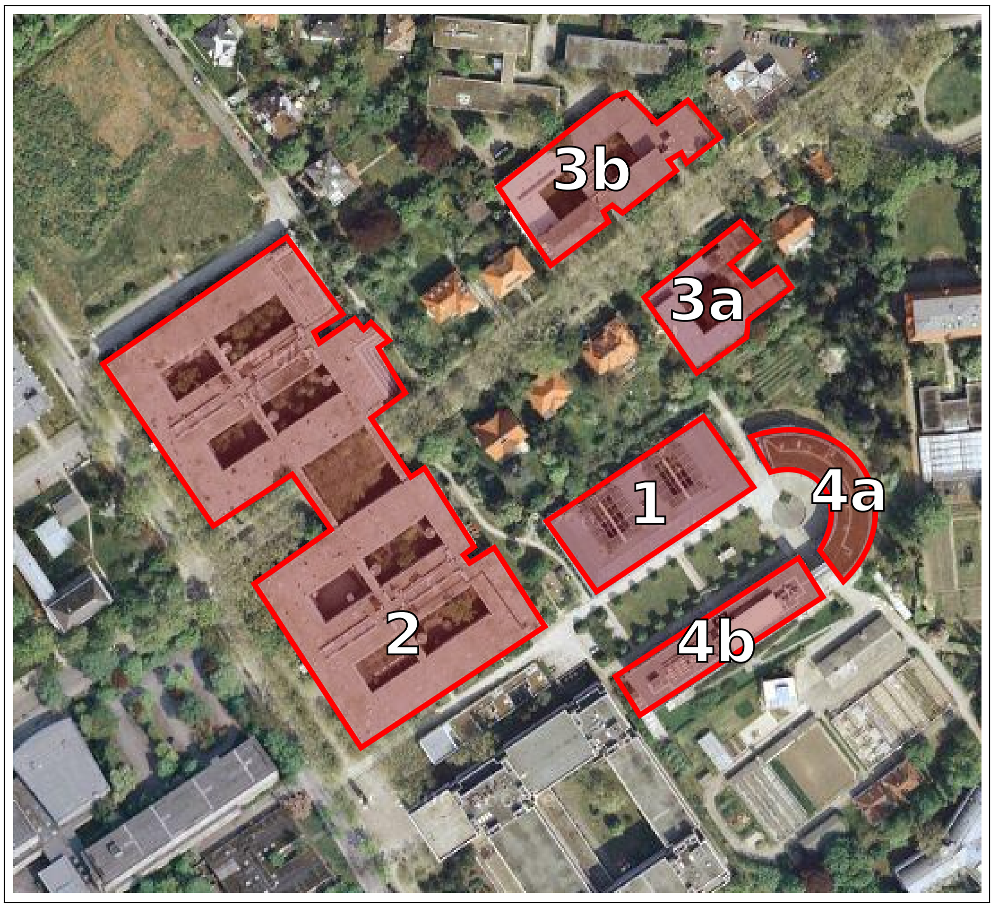


Figure 4.5: Location of the DES testbed in its final setup phase: Computer Science (1), Physics (2), Mathematics (3), and ZIB (4)

- Long linear chain of routers
- Area with high node degree and media access contention among routers
- Regions without coverage creating holes in the network
- Unidirectional connections
- Multiple mesh clouds whose border routers do interfere with each other

These topologies will be configured on demand by switching particular subsets of routers off or by changing their parameters.

The official campus wide ZEDAT managed WLAN called *FUnkLAN* does exist in parallel to the DES testbed using the same wave band. This setup of course leads to additional media contention and possible media congestion. Channel 13 represents an exception as it is not used by the *FUnkLAN* due to compatibility reasons. This is because of NICs of foreign students that might not support this frequency due to frequency restrictions in their home countries. Despite these issues we consider them as normal real world problem experienced in many installations. To ensure repeatability of experiments under equal conditions these will be done at night-shift when no or very few link interruptions due to crowds of people do exist and interfering radio devices are usually not operated.

4.6 Network Interface Card Evaluation

The wireless NICs are the most important part of the DES-Mesh testbed. Several so called chipsets do exist on the market. This term specifies a group of integrated circuits used in one product. In the context of WLAN a chipset refers to the following components:

- RF chip
- Baseband processor
- Media access controller chip
- Bridge chip, like USB to PCI converters

For evaluation purposes we purchased a small number of devices taking care to test at least one of each manufacturer. One of the first arising problems regarding hardware selection has been to obtain technical specification about the available products. Few enduser equipment manufacturers list the used chipset. We even noticed some products did sport different ones depending on the (often not advertised) revision number.

It has been the goal of this evaluation to find a chipset with a corresponding driver (respectively kernel module) that satisfies the following issues:

- The chipset has an open source and working driver.
- USB dongles are supported by the driver.
- The hardware should be as cheap as possible and available on the market even in some years.
- Ad-hoc and monitor mode have to be supported.
- The chipset should be the only component that matters and no manufacturer specific requirements about the USB dongle shall have to be known.
- An active open-source community continuing driver development should exist.
- If the chipset manufacturer provides an open source driver and/or technical specification, that is an important advantage.
- The hardware supports the IEEE 802.11g standard and future upgrades to 802.11n can be done smoothly, requiring no major changes.
- The kernel module should be based on the new unified *mac80211* WLAN stack or one be at least in active development.

Table 4.1 lists an excerpt and the state of the open source kernel modules at the time of the evaluation.

Chipset Manufacturer	Driver	WLAN Modi				Stack	Note
		Managed	Ad-Hoc	Master	Monitor		
Atheros	ath5k*	+	+	-	+	mac80211	no USB support
Atheros	madwifi-ng*	+	+	+	+	??	no USB support
Broadcom	b43, b43legacy*	+	+	+	+	mac80211	no USB support
Prism	ndiswrapper	+	+	-	-	native	kernel panic
Prism	p54usb	??	??	??	??	mac80211	incompatible hardware
Ralink	rt73	+	+	-	+	??	legacy driver
Ralink	rt2x00	+	-	-	+	mac80211	since 2.6.25
Realtek	r8187	+	+	??	+	ieee80211	
Realtek	rtl8187	+	+	??	+	mac80211	
Texas Instruments	acx1xx	??	??	??	??	??	incompatible hardware
Texas Instruments	acx1xx	??	??	??	??	mac80211	incompatible hardware
Texas Instruments	AVM kernel module	+	+	-	-	native	
ZyDAS	ndiswrapper	??	??	??	??	native	kernel panic
ZyDAS	zd1211	+	+	+	-	??	deprecated
ZyDAS	zd1211rw	+	-	-	+	ieee80211	
ZyDAS	zd1211rw	??	??	??	??	mac80211	

Table 4.1: Comparison of the available kernel modules for selected IEEE 802.11 chipsets. Plus symbols mark supported features and minus symbols unsupported ones. Questionmarks represent unknown/untested states.

We have been surprised to experience so many challenges in this process. Atheros based wireless NICs are the de facto standard ones used in research testbeds. Being feature complete and well documented it has been a pity to learn that USB devices are still not supported - neither in the old madwifi nor redesigned ath5k driver[2]. The same applies to Broadcom based NICs. This crucial missing feature did already disqualify two chipsets in the very beginning.

The Prism based stick could not be tested. We suspect a firmware to hardware mismatch although the image has been extracted from the vendor provided driver for Microsoft's Windows OS. We additionally tested several other firmware revisions without success. Because of this uncertainty we refrained from the chipset as we want to avoid such hardware-software dependencies. As we later learned the firmware is required to configure a NET2280 PCI-to-USB bridge as the dongle uses the same parts as found on PCI cards.

Using the native MS Windows driver provided with any product has been out of question. First of all, we cannot review or modify the code and secondly at the time of evaluation ndiswrapper did cause a kernel panic if the EHCI module has been loaded running on our Alix2c2 systems. USB 2.0 of course is prerequisite to use three or more sticks on a single port.

Texas Instrument based hardware is supported limited by the community supplied kernel module. That is too restricting in the face of hardware replaceability. In contrast, the kernel module provided by AVM with their products that are based on TI chipsets did function successfully and perform very well. Sadly the driver does use a very large Binary Large Object (BLOB). Our negotiations with AVM did not result in an agreement to get access to the source code.

For ZyDAS based hardware a vendor provided driver (zd1211) is available that is outdated and is no more developed. The rewritten kernel module does not support the ad-hoc mode and a port to the mac80211 stack has been unusable. Further on, ZyDAS was acquired by Atheros and the future of this product family is unknown.

Left are Realtek and Ralink chipsets. Both feature vendor driver based kernel modules and rewrites for the new mac80211 stack. Even though only the newest revision for Realtek based hardware did already support the ad-hoc mode, we opted for Ralink. We suspect the feature list of all community drivers to harmonize after the port to the unified stack. Until then the legacy Ralink driver will fit our requirements. At the time of the writing Ralink did already offer IEEE 802.11n draft compatible hardware (RT2800 chipset) along with an open source driver. This is, regarding future upgrades, a very promising prospect. All Ralink chipsets have Linux support, not just the one used for evaluation. Our mesh nodes use USB dongles based on the RT2501U [67] architecture (IEEE 802.11b/g) with a RT2571W BB/MAC IC and RT2528 RF IC. The even more interesting RT5201U chipset that supports IEEE 802.11a/b/g could not be found in any end-user product.

In the next step we evaluated which product to use. Hercules HWGUSB2-54, Conceptronic C54RU, and LogiLink WL0025 sticks were purchased. Figure 4.7 shows the internals of the first ones and Figure 4.6 of the LogiLink.

Initial tests showed some noticeable and severe differences. We evaluated the NICs always in pairs. In a first simple test one mesh router was placed in an office of the computer science institute while a laptop was used as mobile node. We used *ping* to track the region in which we still received echo replies. As mentioned in the previous sections some walls of our building have a high attenuation factor. This led to more or less damping

[2]This refers to the Linux driver only as the BSD one does already support USB NICs.

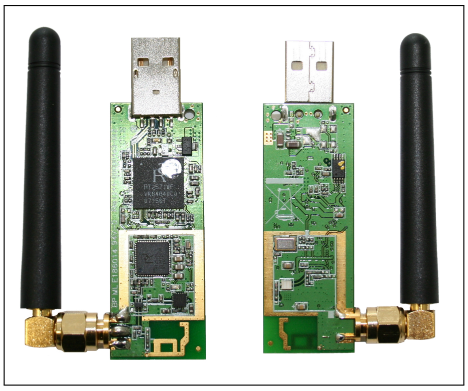
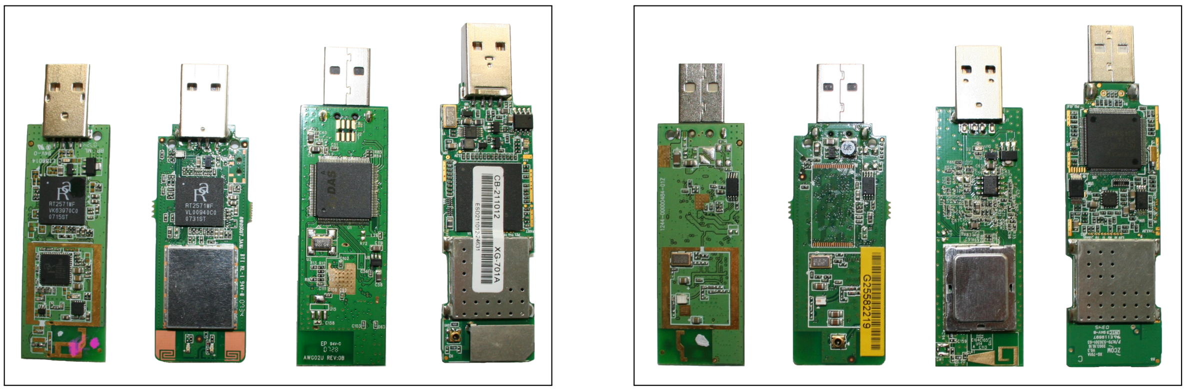


Figure 4.6: LogiLink WL0025 (front and back)



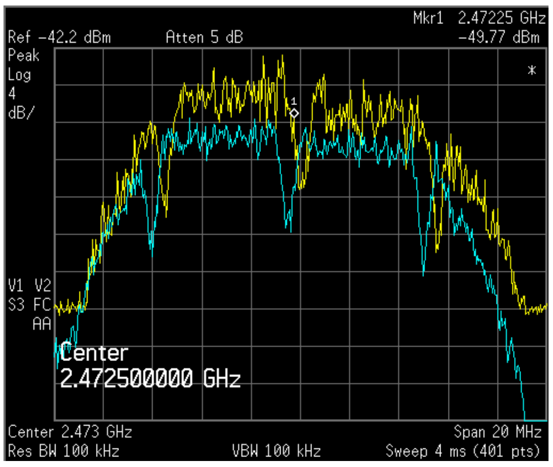
(a) Front (b) Back

Figure 4.7: Wireless NICs (left to right): Hercules HWGUSB2-54, Conceptronic C54RU, Longshine LCS-831G3, Medion MD40900

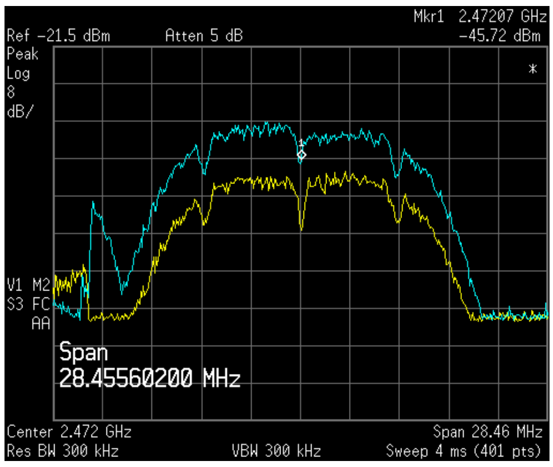
of radio waves depending on the current location. We have been surprised to observe the C54RU having a maximum transmission range of only a few meters. Communication was only possible to the next adjacent office or corridor. The HWGUSB2-54 did perform a little better in this scenario, enabling us to move the mobile node two offices away. Although more satisfactory, we have not been fully convinced by this result. For a high node degree too many routers would have to be placed per area, for example one in every office. The LogiLink WL0025 is equipped with a R-SMA connector plug and is otherwise (on visual inspection) identical in design[3] to the HWGUSB2-54. Using the included 4 dBi Hi-Gain antenna a transmission range of about two times of the last measurement has been noticed. Regarding our building, a placement of routers every two or three offices seems viable.

To cross check our observations we used a spectrum analyzer. Figure 4.8 shows the obtained traces. Each trace is based on a pair of NICs in the same setting. The first experiment using the HWGUSB2-54 and C54RU did confirm our previous subjective impression. Each card was placed in 2 m distance to the spectrum analyzer's antenna and set to channel 13 in ad-hoc mode with a transmission rate of 11 Mbps using DSSS modulation. We generated traffic for 60 s by using the `arping` utility and traced the maximum receive power.

[3]This assertion related to the circuit board layout. Compare Figure 4.6 and Figure 4.7.



(a) HWGUSB2-54 (upper) and C54RU (lower) trace



(b) WL0025 (upper) and HWGUSB2-54 (lower) trace

Figure 4.8: Spectrum Analyzer Traces. Please note the different scaling in both figures.

A difference of about 5 dBm has been measured. The second experiment with the WL0025 and HWGUSB2-54 had to be done using a different test setup compared to the first one as the transmitter position deviated slightly. Nevertheless, comparability between the two tested NICs was retained. About 7 dBm difference in receive power have been measured that we ascribe solely to the antenna as the two dongles seem to be identical in design.

Of course, just the transmit power has been measured in these experiments. The communication range depends on many other factors. The effect of the antenna on the receiver side has also to be considered as well as the amplifier. We did this simple experiment to get an understanding of the observations and to support the statement that it is not sufficient to factor in chipsets only. The entire enduser products have to be evaluated. We conclude that in our setting and according to the goals of the testbed the LogiLink WL0025 is a reasonable choice for our testbed.

CHAPTER 5

Experiment Methodology

Testbeds are primarily constructed to gain a better insight into the operation of algorithms and to do computer network based reseach with a high reality aspect. There are two requirements which have to be considered for comprehensible experiments. First of all, experiments have to be designed so that their measured data can be used for analysis to gain insight. However, this is not a trivial task, as there is no standardized or general approach for such duties. Secondly, experiments have to be designed, described, and conducted in a way which is comprehensible for others and makes these repeatable as well as reusable.

In this Chapter we discuss our methodology concerning the design, description, and execution of experiments. For this we have developed *DES-CRIPT*, an XML-based domain specific language (DSL), that is used to specify experiments.

5.1 Experiment Design

The process of experiment execution in the DES testbed is closely related to its architecture. The testbed server with its management console interface functions as the main control instance and is in charge of all steps of an experiment. Figure 5.1 depicts all steps necessary to run an experiment.

The first task of an experiment is its design and description. Next to some general information such as the experiment's name, start time and duration we divide all tasks in four phases. Each experiment starts with the configuration phase, in which the testbed has to be set up according to the experiments description. This part includes the selection of participating nodes, changes to their configuration, and the distribution of additional software that may include a self compiled kernel modules. Once the testbed is initialized we enter the execution phase. The experiment is executed simultaneously on all participating nodes. The mesh server issues commands to the participating mesh routers at particular times. Therefore, the description provides a feature to define actions integrated into a scheduling system. After the experiment run we progress with the trace phase, in which the mesh router's log files are copied to the testbed server and the results are inserted into the database. We define a specific format for general logging data, so that traces are comparable, reproducible, and can be used as simulation input. In the next step we process the clean up stage, in which the testbed server resets the configuration of the mesh routers and removes optionally added files and software. We also provide a feature to apply user defined evaluation scripts on the gathered log data in order to

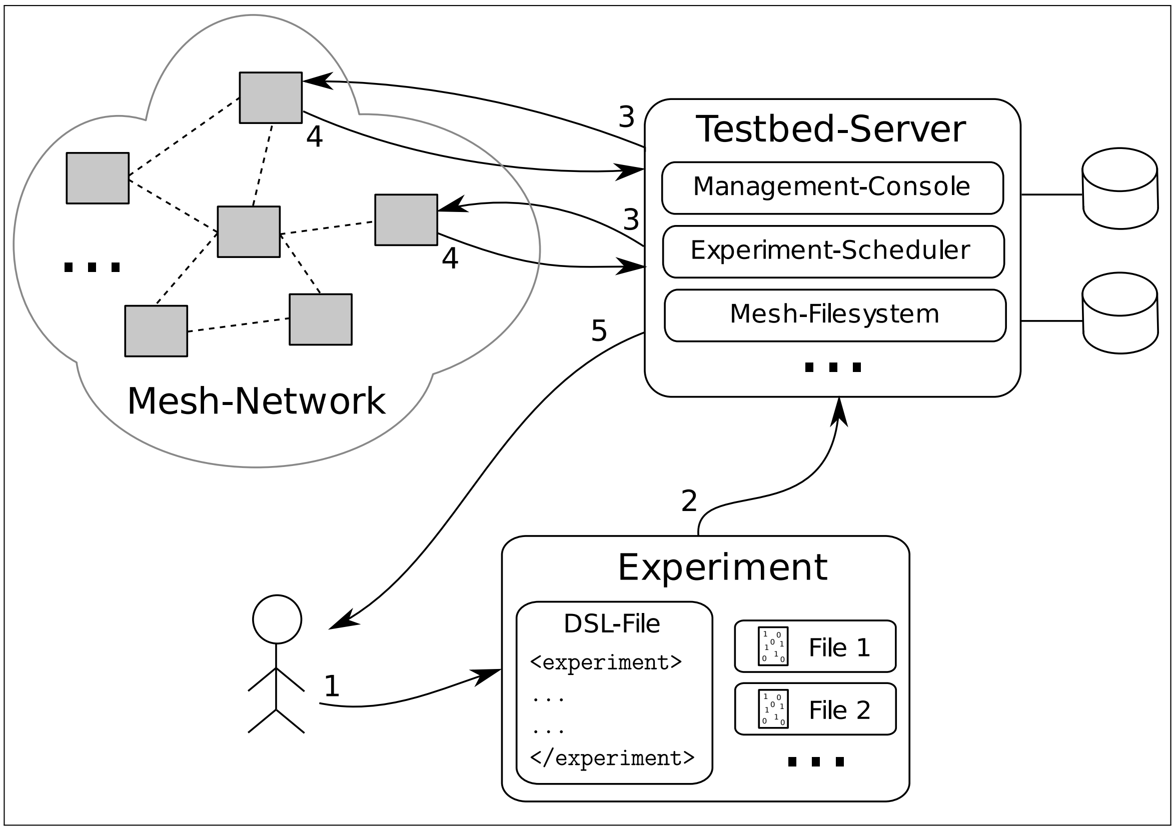


Figure 5.1: Setup and work-flow of an experiment using the DES testbed. In the first step (1) the user creates an experiment package consisting of the *DES-CRIPT* experiment description and several additional files, like a new kernel version. The user uploads the package via the management console to the testbed server (2). The scheduler is hosted at the testbed server. It starts the experiment by logging into the participating nodes that are then configured accordingly (3). The experiment is then executed. Afterwards, all mesh routers copy their log files to the testbed server, which can insert them into the database (4). Finally, the testbed server notifies the user (5) that the experiment has finished and the results are available for further evaluation.

automate as much of the experiment as possible. Finally, the user is notified by an email service that the experiment has finished and that the results are stored on the testbed server for further evaluation.

We developed *DES-CRIPT*, an XML-based DSL for the description of such experiments. *DES-CRIPT* allows to conveniently map the outlined logical structure of an experiment to hierarchical sections. The easy-to-read property of XML satisfies the goal to make experiments comprehensible and easy to understand for other users. Furthermore, we exploit the fact that our testbed management console is implemented using Java, for which a variety of XML parser do exist. Therefore, the integration of the *DES-CRIPT* description parser into the management console represents a straight forward and low overhead task. A sample experiment description file is shown in Listing 5.1. It consists of four sections which are mapped to the outlined hierarchical structure.

```
<experiment>
    <general>
        ...
    </general>

    <init>
        <nodes>...</nodes>
        <files>...</files>
        <database>...</database>
        ...
    </init>

    <run>
        <actions>...</actions>
        ...
    </run>

    <trace>
      ...
    </trace>

    <clean>
        <evaluate>...</evaluate>
        ...
    </clean>
</experiment>
```

Listing 5.1: The structure of an experiment description using *DES-CRIPT*

5.2 Experiment Execution

This section describes how the different steps of an experiment's execution are processed and errors are handled. When an experiment is uploaded, the testbed server interprets the *DES-CRIPT* description and sets the scheduler to start the experiment. The start time is specified in the experiment description but can be adjusted due to the experiment's priority and other scheduled experiments. Once the start time is reached, the execution phase starts. The different stages of the execution phase are depicted in Figure 5.2.

The experiment's execution starts with the *configure stage*, in which the testbed server establishes connections to the participating mesh routers and applies possible modifications to their configuration and file system. Once all nodes are configured, the *testrun stage* begins and the testbed server processes all actions as defined in the *DES-CRIPT* description by sending commands to the mesh routers at the designated times. The *testrun stage* can be repeated as often as specified in the *DES-CRIPT* description file to increase the validity and accuracy of the experiment. By repeating a specific experiment several times, undesired effects caused by changing environmental parameters like the interference level can be weakened.

Subsequently, the *evaluation stage* is processed. In this stage all mesh routers that participated in the experiment transfer their data to the testbed server. The results can be stored in form of log files or can be inserted into the custom database tables as

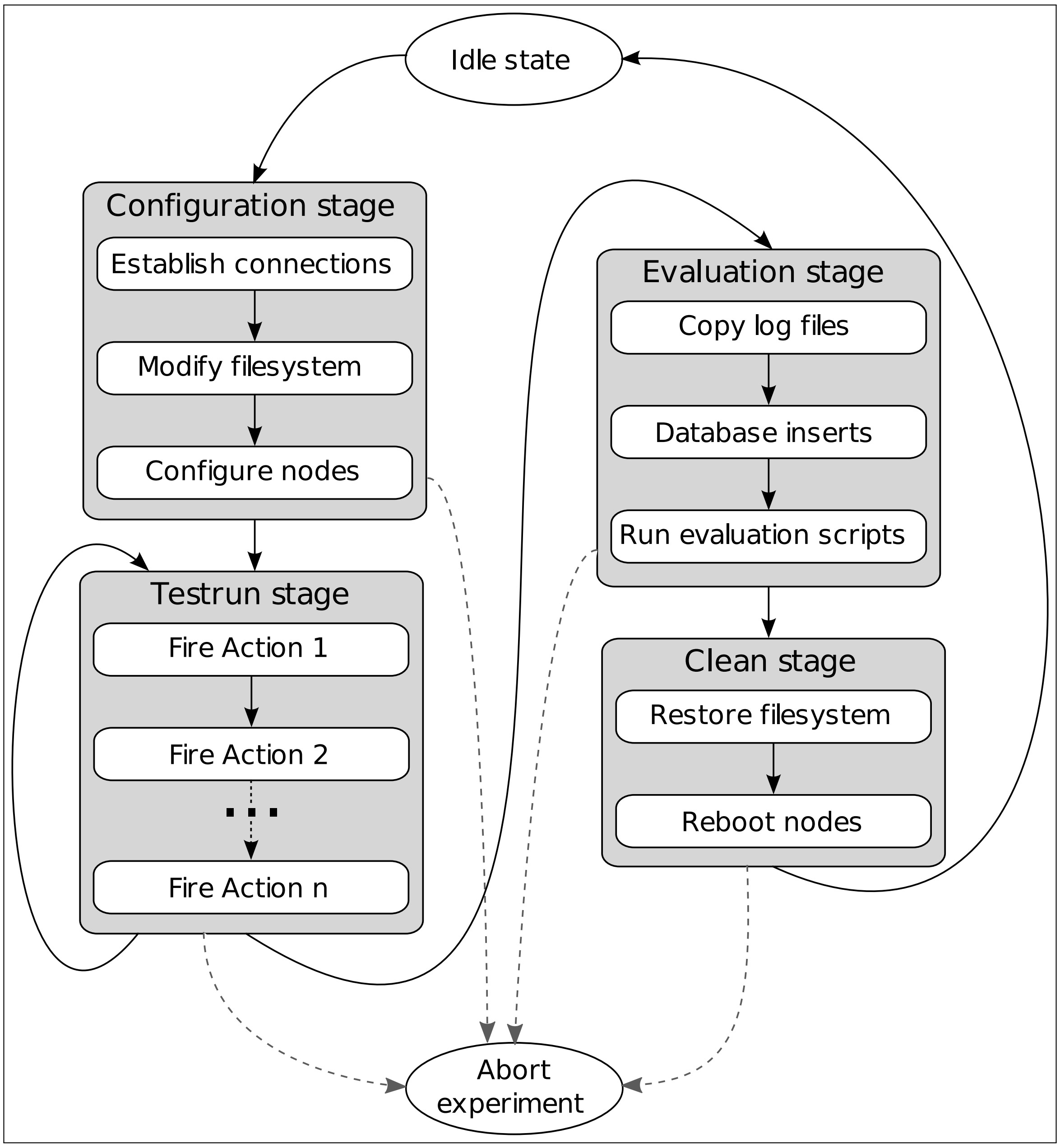


Figure 5.2: Execution phase of an experiment. An experiment run starts with the *configuration stage*, in which the testbed server prepares the participating mesh routers. In the following *testrun stage* the actions for the experiment are executed. This stage can be repeated arbitrary times in order to automatically replicate testruns for achieving better comparable results. Subsequently in the *evaluation stage*, the results are stored on the testbed server. Finally the *clean stage* is processed, in which the testbed server removes experiment specific files from the mesh routers and restores their configuration.

specified in the *DES-CRIPT* description. General data of each experiment are saved in the database by default, for example its duration and repetitions. The last step of this stage consists of the execution of user supplied evaluation scripts on the gathered logged data if specified in the *DES-CRIPT* description. Finally, in the *clean stage* the testbed server removes all files which were added to the file system of the mesh routers and restores their configuration. Additionally, the participating mesh routers can be rebooted, thus all traces of the current experiment are removed from the testbed and the next experiment can be executed in a pristine environment.

During the execution of an experiment errors and undefined behavior can occur. These will most likely affect the results of the experiment and therefore need special treatment. In our testbed error handling is taken care of in a conservative manner. This means that the testbed server aborts an experiment in case of errors or if undefined behavior is detected. We chose this approach to assure validity and comparability of the measured data. The latter can only be ensured if the setup and state during execution of different experiment repetitions is always the same. The error handling mechanism is active in all stages of the experiments execution phase. In the setup stage for instance, the testbed server will already abort the experiment, if no connection can be established with one or more of the participating mesh routers. Depending on the type of the error, it is optional to automatically reschedule the failed experiment. In any case, the user will be notified about the abort with detailed information of the unsuccessful experiment.

5.3 Hints for Experiment Evaluation

The evaluation of network experiments is not an easy task. No simple instructions or reliable information do exist how valid and reproducible results can be achieved. This fact is due to the various testbeds with their different architectures and research objectives and because run experiments usually differ in a significant way that makes it impossible to introduce such a general approach. Still, some guidelines can be set up to simplify the evaluation process and ensure a certain degree of validity and reproducibility.

The DES testbed will be used primarily for research done by students and work group staff. It will be helpful for all users to answer the following questions for each experiment that is performed. The questions represent our point of view and are meant as an advice for consideration mainly for students.

- Questions concerning the set up and execution:
  - What is the number of experiment repetitions?
  - How long does the experiment take?
  - What is the experiment's start / end time?
  - Which nodes were involved? Which changes were made to their configuration?
- For each measurement, the following performance statistics should be calculated if possible:
  - Average values
  - Standard deviation
  - Variance
  - Confidence Interval (CI)

5.4 Tools

In this section we provide a list of the most common tools that are mainly used for experimentation in the execution phase. This shall introduce these to users new to the domain of computer networks. As we already explicated in this chapter, experiments are done fully automated at arbitrary times. The XML-based *DES-CRIPT* description defines actions for the different steps. These actions are small commands executed by the testbed server on the corresponding mesh router(-s). The commands may make use of the following utility programs that are included in our customized Linux distribution among others. Additional software will be installed on demand but it is our goal to refrain from custom developed tools in the favor to use standard software whenever possible. By using this approach we want to maintain repeatability of experiments and verification of their data using arbitrary systems either by ourselves or other members of the research community.

5.4.1 Network Management

Network management tools have the task to change the parameters of routers to modify network properties. In addition some of them provide testing and logging features to probe the network.

***netstat*:** This tool prints routing tables, network connections, masquerade connections, interface statistics, and multicast memberships. *netstat* cannot display the additional routing tables used for policy routing.

***nmap*:** This tool scans hosts or whole networks. It can discover the services running on a host, the OS, firewalls, and many other properties. While sometimes used to scan for exploitable security flaws by attackers *nmap* is the tool of choice for security audits by network administrators.

***route & ip*:** *route* can be used to manipulate the IP routing table and show its contents. *ip* is more advanced as it will allow manipulation and display of all 256 routing tables present if policy routing is done. *ip* is also able to add and remove rules. Other features include displaying and flushing of the route cache.

Examples:

- show route cache: `route -C -n`
- clear route cache: `ip route flush cache`
- show neighbors: `ip neigh show`

***traceroute & mtr*:** Using IP's time to live (TTL) field *traceroute* attempts to track the route packets take through the network to a destination address. Internet Control Message Protocol (ICMP) time exceeded messages are provoked and thus each hop of the path is discovered. Not all hosts send these messages as it is not implicitly required by ICMP. Therefore, Transmission Control Protocol (TCP) SYN probes can also be used. The tool *mtr* is a combination of *traceroute* and *ping* which also measures the round trip delay to each hop of the path from the source.

***tcpdump*:** Network monitoring can be done using *tcpdump* [71] by printing packet headers and their content. Filtering facilities (boolean expressions) enable the user to reduce the amount of logged data. By capturing untruncated packets written to a file Wireshark [72] (formerly Ethereal) can be used as a viewer.

***nast*:** This tool is very similar to *tcpdump*. Packets can be sniffed in the promiscuous mode and various filters applied. Many interesting features are provided in addition, like mapping the LAN by performing Address Resolution Protocol (ARP) requests.

***netcat*:** This program reads and writes data across network connections. TCP and User Datagram Protocol (UDP) may be used as transport layer protocols.

***Kismet*:** Kismet [73] is a combined IEEE 802.11 layer 2 wireless network detector, sniffer, and intrusion detection system. The Kismet Log Viewer (KLV) [74] can be used to generate HTML output of Kismet log files.

***EtherPuppet*:** The above mentioned tools have to be executed on the corresponding host. If we like to omit the log-in step, a tunnel can be created by *EtherPuppet* [75] from the host to another entity. The startpoint of the tunnel is any physical network interface while the endpoint is a virtual one. TCP is used as transport layer protocol to ensure all packets are tunneled successfully. Thus, tools like *tcpdump* can be used on the virtual interface like on the real one of the corresponding host.

5.4.2 Benchmarking

To measure network performance and apply throughput or delay as a metric for routing algorithms these tools can be used.

***ping* & *arping*:** The well-known *ping* command sends `ICMP_ECHO_REQUEST`s and waits for `ICMP_ECHO_REPLY`s to test if a host is reachable. The *arping* program is similar to ping, but uses ARP. The usage of *arping* often results in better reachability, since the packets are smaller which makes the transmission process less error-prone due to the high bit error rate of the wireless medium.

***bing*:** By using `ICMP_ECHO_REQUEST`s the throughput between two hosts is guessed.

***iperf*:** *iperf* is a tool to measure TCP and UDP bandwidth performance. It allows the tuning of various parameters and characteristics.

***netperf*:** The network performance benchmark tool supports throughput and end-to-end latency tests.

***TTCP*:** Various programs based on the original *TTCP* or implemented to mimic its feature set do exist, like *nTTCP* and *nuTTCP*. Their functionality is comparable to *iperf*.

5.4.3 Spoofing & Injection & Packet Creation

This set of tools can be used for more fine grained and detailed network measurements and tests. By assembling custom packets operating systems, firewalls, or general daemons providing a service can be audited.

***Packit*:** Packit (**Pack**et tool**kit**) [76] is a network auditing tool. In general, Packit can customize, inject, monitor, and manipulate IP traffic. Also, lots of TCP, UDP, ICMP, IP, ARP, Reverse Address Resolution Protocol (RARP), and Ethernet protocol header options can be modified. This is useful to simulate network traffic and test firewall settings.

***hping*:** This tool can be used to assemble TCP/IP packets and analyze hosts.

***nemesis*:** Utility with the goal to provide an easy to use text based interface for packet assembly and injection by manual or scripted means.

CHAPTER 6

Conclusion

With the first draft, the intention of this paper was to offer a compact introductory material for students who want to do their thesis in our working group. It became obvious that clarifying the terminology alone does not provide enough information and therefore we added some material about related work. In order to show the difference of our approach with existing testbeds we added a chapter about our ongoing work. Many of our students will have to perform some kind of experiments in the testbed and evaluate their data. Since the manual execution and accomplishment of experiments are extremely labor-intensive and many pitfalls do exist, we thought about some formal method to simplify this task. The resulting chapter introduces such an approach and discusses also some performance metrics with statistical evaluation. Our report is continued in a second publication [9] with a focus on the physical and technical aspects of the DES testbed.

Bibliography

[1] M. Weiser, "The computer for the twenty-first century," *Scientific American*, vol. 265, no. 3, pp. 94–104, 1991.

[2] U. Hansmann, L. Merk, M. S. Nicklous, and T. Stober, *Pervasive Computing Handbook*, 1st ed. Springer-Verlag, 2001, iSBN: 3-540-67122-6.

[3] S. Kurkowski, T. Camp, and M. Colagrosso, "Manet simulation studies: the incredibles," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 9, no. 4, pp. 50–61, 2005.

[4] M. GÃijnes and M. Wenig, *Models for Realistic Mobility and Radio Wave Propagation for Ad-hoc Network Simulations*. Springer, 2009. [Online]. Available: http://www.springer.com/computer/communications/book/978-1-84800-327-9

[5] "The ns and nam web page." [Online]. Available: http://www.isi.edu/nsnam/

[6] "Omnet++." [Online]. Available: http://www.omnetpp.org/

[7] "Homepage of scalable network technologies." [Online]. Available: http://www.scalable-networks.com/

[8] "Opnet modeler." [Online]. Available: http://www.opnet.com/

[9] M. Güneş, B. Blywis, F. Juraschek, and P. Schmidt, "Practical issues of implementing a hybrid multi-nic wireless mesh-network," Freie Universität Berlin, Tech. Rep., 2008.

[10] H. Lundgren, D. Lundberg, J. Nielsen, E. Nordström, and C. Tschudin, "A large-scale testbed for reproducible ad hoc protocol evaluations," in *Proc. WCNC2002 Wireless Communications and Networking Conference 2002 IEEE*, vol. 1, 2002, pp. 412–418 vol.1.

[11] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks*, vol. 47, no. 4, pp. 445–487, March 2005. [Online]. Available: http://www.sciencedirect.com/science/article/B6VRG-4F53V5H-2/2/9fa1587e47665f1fb3f7fb461461dd6b

[12] Carleton University, "Wireless mesh networking," http://kunz-pc.sce.carleton.ca/MESH/index.htm.

[13] A. Raniwala, A. Raniwala, and T.-c. Chiueh, "Architecture and algorithms for an ieee 802.11-based multi-channel wireless mesh network," in *Proc. IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2005*, T.-c. Chiueh, Ed., vol. 3, 2005, pp. 2223–2234 vol. 3.

[14] A. Raniwala, K. Gopalan, and T. cker Chiueh, “Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks,” *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 8, no. 2, pp. 50–65, 2004.

[15] S. M. Das, D. Koutsonikolas, and Y. C. Hu., *Handbook of Wireless Mesh and Sensor Networking*. McGraw-Hill International, 2007, ch. Measurement-based Characterization of a Wireless Mesh Network.

[16] “Microsoft research - mesh connectivity layer.” [Online]. Available: http://research.microsoft.com/mesh/

[17] D. B. Johnson and D. A. Maltz, “Dynamic source routing in ad hoc wireless networks,” in *Mobile Computing*, T. Imielinski and H. Korth, Eds. Kluwer, 1996, vol. 353.

[18] R. Draves, J. Padhye, and B. Zill, “The architecture of the link quality source routing protocol.” Microsoft Research, Tech. Rep., 2004.

[19] P. De, A. Raniwala, S. Sharma, and T. Chiueh, “Mint: a miniaturized network testbed for mobile wireless research,” in *Proc. IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2005*, vol. 4, 2005, pp. 2731–2742 vol. 4.

[20] P. De, A. Raniwala, R. Krishnan, K. Tatavarthi, J. Modi, N. A. Syed, S. Sharma, and T. cker Chiueh, “Mint-m: an autonomous mobile wireless experimentation platform,” in *MobiSys ’06: Proceedings of the 4th international conference on Mobile systems, applications and services*. New York, NY, USA: ACM, 2006, pp. 124–137.

[21] B. A. Chambers, “The grid roofnet: A rooftop ad-hoc wireless network,” Master’s thesis, Massachusetts Institute of Technology, 2002. [Online]. Available: http://pdos.csail.mit.edu/roofnet/doku.php?id=centralsq

[22] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, “The click modular router,” *ACM Transactions on Computer Systems*, vol. 18, no. 3, pp. 263–297, August 2000. [Online]. Available: http://www.read.cs.ucla.edu/click/publications

[23] “Berlin roofnet project.” [Online]. Available: http://sarwiki.informatik.hu-berlin.de/BerlinRoofNet

[24] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh, “Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols,” in *Proc. IEEE Wireless Communications and Networking Conference*, vol. 3, 2005, pp. 1664–1669 Vol. 3.

[25] J. Lee, S. L. Shrestha, K. Lee, J. Lee, A. Lee, and S. Chong, “Practical guidelines for implementing a wmn testbed: From design to deployment,” submitted to IEEE Network Magazine.

[26] S. L. Shrestha, J. Lee, A. Lee, K. Lee, J. Lee, and S. Chong, “An open wireless mesh testbed architecture with data collection and software distribution platform,” in *Proc. 3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities TridentCom 2007*, J. Lee, Ed., 2007, pp. 1–10.

[27] "Freifunk." [Online]. Available: http://start.freifunk.net/

[28] "Freimap." [Online]. Available: http://relet.net/trac/freimap

[29] Colorado School of Mines, "Center for automation, robotics, and distributed intelligence." [Online]. Available: http://egweb.mines.edu/cardi/

[30] M. Dyer, J. Beutel, L. Thiele, T. Kalt, P. Oehen, K. Martin, and P. Blum, "Deployment support network - a toolkit for the development of wsns," in *Proc. 4th European Conf. on Wireless Sensor Networks (EWSN 2007)*. Springer, Berlin, Jan. 2007, pp. 195–211.

[31] A. Arora and et al, "Exscal: elements of an extreme scale wireless sensor network," in *Proc. 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, R. Ramnath, Ed., 2005, pp. 102–108.

[32] A. Arora, E. Ertin, R. Ramnath, M. Nesterenko, and W. Leal, "Kansei: A High-Fidelity Sensing Testbed," *IEEE Internet Computing*, vol. 10, no. 2, pp. 35–47, 2006.

[33] G. Werner-Allen, P. Swieskowski, and M. Welsh, "Motelab: a wireless sensor network testbed," in *Proc. Fourth International Symposium on Information Processing in Sensor Networks IPSN 2005*, 15 April 2005, pp. 483–488.

[34] University of Southern California, "Tutornet: A tiered wireless sensor network testbed." [Online]. Available: http://enl.usc.edu/projects/tutornet/

[35] V. Handziski, A. Köpke, A. Willig, and A. Wolisz, "Twist: A scalable and reconfigurable wireless sensor network testbed for indoor deployments," Technical University Berlin, Tech. Rep., 2005.

[36] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. New York, NY, USA: ACM, 2002, pp. 88–97.

[37] R. Szewczyk, J. Polastre, A. Mainwaring, and D. Culler, "Lessons from a sensor network expedition," in *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, Jan. 2004. [Online]. Available: citeseer.ist.psu.edu/szewczyk04lessons.html

[38] F. Heil, C. Hellmich, B. Guilford, Schiller, and Naumowicz, "Autonomous monitoring of vulnerable habitats using a wireless sensor network," in *Proceedings of the Workshop on Real-World Wireless Sensor Networks*, 2008.

[39] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, "Hardware design experiences in zebranet," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2004, pp. 227–238.

[40] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with zebranet," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. 5, pp. 96–107, 2002.

[41] K. Mayer, K. Ellis, and K. Taylor, "Cattle health monitoring using wireless sensor networks," in *ICCCN*, 2004.

[42] UC Davis Wildlife Health Center, "Southern california puma project," 2004. [Online]. Available: www.vetmed.ucdavis.edu/whc/scp/

[43] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2005, pp. 51–63.

[44] K. Smettem, M. Kranz, R. Cardell-Oliver, and K. Mayer, "Field testing a wireless sensor network for reactive environmental monitoring," in *Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, 2004.

[45] ——, "A reactive soil moisture sensor network: Design and field evaluation," in *International Journal of Distributed Sensor Networks*, 2005, vol. 1, no. 2, ch. 149 ÃŕÂ£ÅŞ 162.

[46] Colorado School of Mines, "Edgar mine sensornet." [Online]. Available: http://ore.mines.edu/~mcolagro/edgarmine/live-readings

[47] G. Werner-Allen, K. Lorincz, J. Johnson, J. Lees, and M. Welsh, "Fidelity and yield in a volcano monitoring sensor network," in *OSDI '06: Proceedings of the 7th symposium on Operating systems design and implementation*. Berkeley, CA, USA: USENIX Association, 2006, pp. 381–396.

[48] K. Martinez, A. Riddoch, J. K. Hart, P. Padhy, and H. L. R. Ong, "Glacial environment monitoring using sensor networks," in *Proceedings of the First Real-World Wireless Sensor Networks Workshop (REALWSN'05)*, 2005, pp. 10–14.

[49] H. Paulsen, "PermaSensorGIS - Real-Time Permafrost Data Collection," *Geoconnexion International Magazine*, Feb 2008. [Online]. Available: http://www.sensorgis.de/cms/index.php?option=com_content\&task=view\&id=48\&Itemid=42

[50] "PermaSensorGIS - website of the project." Apr 2008. [Online]. Available: http://www.sensorgis.de

[51] C. Renner, M. Venzke, S. Waschik, C. Weyer, V. Turau, and M. Witt, "The heathland experiment: Results and experiences," in *Proceedings of the REALWSN'05 Workshop on Real-World Wireless Sensor Networks*, 2005. [Online]. Available: citeseer.ist.psu.edu/731645.html

[52] "Meraki." [Online]. Available: http://meraki.com/

[53] "Mit roofnet." [Online]. Available: http://pdos.csail.mit.edu/roofnet/doku.php?id=centralsq

[54] "Belair networks." [Online]. Available: http://www.belairnetworks.com/

[55] "firetide." [Online]. Available: http://www.firetide.com/

[56] "Motorola meshnetworks." [Online]. Available: http://www.motorola.com/mesh/

[57] "Tropos." [Online]. Available: http://www.tropos.com/

[58] "Packethop." [Online]. Available: http://www.packethop.com/

[59] "Warp." [Online]. Available: http://warp.rice.edu/news.php

[60] "Gnu radio." [Online]. Available: http://www.gnu.org/software/gnuradio/

[61] A. Taherkordi, M. A. Taleghan, and M. Sharifi, "Achieving availability and reliability in wireless sensor networks applications," *ares*, vol. 0, pp. 529–535, 2006.

[62] K. Pawlikowski, "Do not trust all simulation studies of telecommunication networks," in *ICOIN*, 2003, pp. 899–908.

[63] B. Blywis, F. Juraschek, M. Güneş, and J. Schiller, "Design concepts of a persistent wireless sensor testbed," in *7. GI/ITG KuVS Fachgespräch "Drahtlose Sensornetze"*, 2008.

[64] M. Baar, H. Will, B. Blywis, A. Liers, G. Wittenburg, and J. Schiller, "The scatterweb msb-a2 platform for wireless sensor networks," Freie Universität Berlin, Tech. Rep., 2008.

[65] P. Fuxjager, D. Valerio, and F. Ricciato, "The myth of non-overlapping channels: interference measurements in ieee 802.11," in *Proc. Fourth Annual Conference on Wireless on Demand Network Systems and Services WONS '07*, 2007, pp. 1–8.

[66] "Alix2 / alix3 series manual." [Online]. Available: http://www.pcengines.ch/pdf/alix2.pdf

[67] "Ralink technology, Ralink RT2501U chipset product specification." [Online]. Available: http://www.ralinktech.com/

[68] Freie Universität Berlin, "Scatterweb project." [Online]. Available: http://cst.mi.fu-berlin.de/projects/ScatterWeb/

[69] A. Zimmermann, M. Güneş, M. Wenig, J. Ritzerfeld, and U. Meis, "A hybrid testbed for wireless mesh networks," in *Proceedings of the 1st International Workshop on Operator-Assisted (Wireless Mesh) Community Networks (OpCommâĂŹ06)*. IEEE Communications Society Press, September 2006.

[70] "One laptop per child (olpc)." [Online]. Available: http://www.laptop.org/

[71] "tcpdump/libcap." [Online]. Available: http://www.tcpdump.org/

[72] "Wireshark." [Online]. Available: http://www.wireshark.org/

[73] "Kismet." [Online]. Available: http://www.kismetwireless.net/

[74] "Kismet log viewer." [Online]. Available: http://www.mindflip.org/klv/

[75] P. Biondi, "Etherpuppet." [Online]. Available: http://www.secdev.org/projects/etherpuppet/

[76] "Packit (packet toolkit)." [Online]. Available: http://www.packetfactory.net/projects/packit/